

THE ART OF PCB REVERSE-ENGINEERING

Ng Keng Tiong

VISIT...

LANZAROTE
Caliente.COM

Copyright © 2015 by Ng Keng Tiong. All rights reserved.
Cover designs (front and back) by the author.

Screenshots from Microsoft® Visio are used with permission from Microsoft. Both Microsoft and Visio are registered trademarks of Microsoft Corporation in and outside the United States. Products and service mentioned in this book are trademarks or registered trademarks of their respective companies. All trademarks and registered trademarks in this book are the property of their respective holders.

No part of this book may be reproduced in any form, or stored in a database or retrieval system, or transmitted or distributed in any form, by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the author.

LIMIT OF LIABILITY AND DISCLAIMER OF WARRANTY

The information, examples, illustrations, documentation, and other references in this book are provided "as is", without warranty of any kind, expressed or implied, including without limitation any warranty concerning the accuracy, adequacy, or completeness of the material or the results obtained from using the material. Neither the publisher nor the author shall be responsible for any claims attributable to errors, omissions, or other inaccuracies in the material in this book. In no event shall the publisher or author be liable for direct, indirect, special, incidental, or consequential damages in connection with, or arising out of, the construction, performance, or other use of the materials contained herein.

Print copy ISBN-13: 978-1499323443

DEDICATION

This book is dedicated to all electronics enthusiasts who share the same passion and appreciation for printed circuits and possess the curiosity and tenacity of engineers to unravel the beauty of the original design.

CONTENT

	Acknowledgment	7
	Preface	9
1	Before You Begin	17
	What You Need to Know	18
	Why Bother?	19
	My Personal Story	20
	Legal and Copyright Issues	25
	The Three D's	27
2	Basic Preparation Work	31
	Getting to Know Your PCB	31
	Accessibility	32
	Bill of Materials	33
	Resistors and Networks	35
	Capacitors	37
	Inductors and Transformers	40
	Fuses	42
	Relays and Switches	43
	Crystals and Oscillators	45
	Diodes and Zeners	46
	Transistors and MOSFETs	48
	Integrated Circuits	49
	Miscellaneous	54
	Components without Markings	55
	Missing Reference Designators	56
	Filling up the BOM	58
	Conformal Coating	60
	Component Datasheets	62

3	Laying Out The PCB Artwork	67
	Is it Really Necessary?	68
	Getting to Know Microsoft Visio	69
	Initial Visio Settings	70
	Keyboard Shortcuts	76
	Quick Maneuvering Tricks	76
	Manipulating Objects	78
	A Simple Example	80
	Preparing the Drawing Page	81
	Drawing the PCB Outline	86
	Creating Component Layout Symbols	93
	Populating the PCB	102
	A Complex Example	106
	More About Circuit Boards	108
	PCB-Mount Fixtures	109
	Bolts and Nuts	111
	Connectors	112
	Toggle Switches	115
	Complex IC Layout Symbols	116
	Drawing Analog Components	121
	Mobbed!	127
	Too Small to Ignore	130
 4	 Wiring Up The Schematics	 135
	Spaghetti, Anyone?	136
	Basic Tools	136
	ANSI or IEC?	141
	Color or Monochrome?	142
	Hierarchical or Flat?	144
	Elements of a Good Schematic Diagram	145
	Drawing Schematic Symbols	151
	Approach and Strategy	175
	It's an Analog World!	188
	An ATX Power Supply Example	189
	Strategy for Analog PCBs	192
	One Useful Trick	197
	A Word of Caution	198
	Doing the Unthinkable	199

5	Advanced Topics	203
	Layering Technique	203
	Simplicity in Complexity	219
	Generating Bill of Materials	257
	The Matrix	263
	Summary	270

6	Going From Here	273
	Fabulous Freebies	274
	Web-Based Wonders	277
	Old-but not Obsolete	280
	Commercial Suites	281
	Rantings on Circuit Simulation	286
	Summary	287

Appendix A Electronics References

Appendix B Visio References

Appendix C PCB Layout

Appendix D Schematic Entry

Appendix E Advanced Topics

Bibliographies

Index

ACKNOWLEDGMENTS

This book would not have been possible without the understanding and support of my wife, [Bernice](#), when I expressed to her my desire to write a book of this nature that is totally outside her knowledge domain.

Thanks to [CreateSpace](#) for making it easy for aspiring authors to publish and distribute their works with its print-on-demand approach, removing all the hassles and obstacles which would otherwise severely limit the availability or prevented the circulation of this book.

PREFACE

I was introduced to Electronics at the age of 15 in my higher secondary school days. I was one of those fortunate batches of students—in fact, the last in-take to study basic electricity and electronics in the school's curriculum in 1978.

Electronics was totally new to me, and I had problem understanding some of the basic concepts back then. It might surprise you that I had trouble figuring out the milli-, micro-, nano-, Kilo- and Mega- prefixes in Ohm's Law, struggling with corrections and re-corrections in my class assignments I almost gave up the subject for fear and frustration. Thankfully the light bulb turned on in me after a harrowing first semester, and from then on there was no looking back as I went on to do well for my final exams and did my major in Electronics and Communication for my tertiary education.

Due to family financial difficulties, I signed up with the Republic of Singapore Air Force (RSAF) after my graduation as an aircraft radar and communications (RC) technician. I did well in the Air Engineering Training Institute (AETI) and was selected for the E-2C Hawkeye program. In May of 1986, I attended a six-month training at the Grumman Aerospace Corporation's premise in Long Island, New York. There for the first time, I was introduced to the concept of automated test systems (ATS) as I learned how to operate and maintain the CAT-IIID and RADCOM test stations, and used these awesome equipment to perform test diagnostics on 75% of the E-2C's sophisticated avionics (radar, communications, navigation, displays, power supplies, etc.)

My invaluable experience as one of the pioneers in setting up the E-2C squadron's third-line repair bay and running the daily maintenance operation, as well as training three batches of local technicians, laid a solid foundation for my engineering career in electronics, so much so that I was head-hunted and invited to join Singapore Technologies (ST) Electronics Limited, a subsidiary of Singapore's home-grown defense industry, ST Engineering Limited, right after my first contract with the RSAF expired. I've been with the company since for over 23 years and now a Principal Engineer by title.

Learning is a life-long process. The same is true in electronics and in the field of test engineering, which is still my primary job scope and interest. I've worked on a variety of printed circuit boards (PCBs) from through-holes using the humble TTL logic circuits, to the high-density surface-mount type multi-layered boards containing complex VLSI, FPGA and ASIC BGA chips. The rate at which integrated circuits grow in complexity and density is staggering, especially in the last decade or so. Conventional way of testing PCBs using in-circuit testers no longer seem practical or even feasible. Functional approach has also become more challenging and is reaching a saturation point to be considered viable. The way to go now would be boundary-scan (JTAG) testing for PCBs designed to take advantage of this technology, though at the expense of chip and board real-estates, or hot-bed embedded testing if design specifications are available for such implementation.

Ranting aside. The fact that you picked up this book inherently suggests that you have an interest (or at least curiosity) in the topic of PCB reverse engineering. In this book, I will share what I've learned in my years of working on PCBs with no schematic diagrams, my approach of assessing a PCB, analyzing

and reconstructing its electrical connectivity, the tools and methods I employed, useful tips to take note, as well as pitfalls to avoid. Sprinkled in the footnotes you will find some interesting anecdotes and personal takes to keep the subject of this book light-hearted and (hopefully) engaging.

WHAT THIS BOOK IS ABOUT

This book will not teach you to become proficient in using electronic design automation (EDA) tools or software to produce or re-produce PCBs from schematic entry to gerber design files. It is not even intended to give you a formal study on PCB structural design and fabrication. There are many books out there that already addressed these topics and by authors who are certainly more knowledgeable and the authority in these fields.

That said, this book does impart knowledge on PCBs that relate to the subject of reverse engineering, as a basic understanding of any PCB you intend to reverse engineer is essential to your success. Also, I will be using Microsoft® Visio albeit in a generic manner to demonstrate the steps involved. I said 'generic' because I'm not advocating any particular version or release of this diagramming tool, which was originally sold and distributed by Shapeware in 1992 (subsequently called Visio Corporation in 1995 when it went public listed), but was acquired by Microsoft Corporation in 2000.

HOW TO USE THIS BOOK

This book is written for easy reading. While you may choose to read selectively since each chapter is a self-contained unit, you are encouraged to go through the book sequentially to derive the maximum benefit. I suggest you read Chapter 1 before launching out to explore the other chapters. A summary of the chapters are listed below:

- Chapter 1 I know you're eager to get started right away, but it's good to know a little of what you're getting into before you plunge in completely. You'll need to have certain background in electronics. I'll mention some available alternatives to doing reverse engineering completely by hand, though they're not necessarily affordable unless you have deep pockets and are willing to part with your hard-earned cash. I'll also share my personal story on what started me on this journey (hopefully a good and inspiring read to you). Then there's the copyright issue which I'll give my personal take on it, before rounding up the chapter with a little work philosophy.
- Chapter 2 This chapter may be thought of as a practical introduction or revision to electronics in terms of the various building blocks of a PCB—resistors, capacitors, inductors, fuses, relays, diodes, zeners, transistors, MOSFETs, and ICs, etc. You may think it's boring or too elementary but I've put in quite a substantial amount of information here, so you should be able to find some rare gems among the junks (to borrow a phrase from the PCB recycle industry). I'll deal with components without markings and reference designators, touch on a bit about conformal coatings, as well as how component datasheets will be a great help later on. These elements formed what I call the ABCD's of preparation work—an essential step that will save you a lot of time and trouble when you start to do the real work.

- Chapter 3 We'll get down to serious business and start doing some interesting PCB layouts and components beginning with this chapter. I will provide a brief introduction to Microsoft Visio, the diagramming software we'll be using throughout the book. You will learn how to setup and navigate the Visio workspace, familiarize yourself with its various tools and functions, decide on which template to use or begin from scratch, create your own component layouts in relation to their datasheets, and more. I have included some of my very own PCB artworks to motivate you and show you what you can possibly do with Visio.
- Chapter 4 Things will get more exciting as we begin to draw the schematics of a PCB. Besides showing you how to create the various component symbols and elements of a schematic, I will also teach you the strategic approaches to reverse engineering different types of PCBs (digital, analog, power, etc.). At the same time, you will also learn why it is advantageous to have a PCB layout and BOM in the first place. Having a good sense of proportion and direction is important, especially if your schematic diagram is going to span multiple pages; I will lay down specific guidelines so you know how to plan your schematics for better readability and consistency.
- Chapter 5 In this chapter, I will show you some advanced techniques you can apply to achieve a more professional level in creating both the PCB layout and schematic diagram. One word of caution: attaining mastery of anything does come with a price tag (Einstein would agree with that, being one of the greatest minds in the twentieth century). If you're willing to put in the extra efforts, I guarantee that you'll not only excel in the art of PCB reverse engineering, you'll pick up a life skill—an attitude—that will enable you to go farther, and a new found confidence to face challenges in your engineering career.
- Chapter 6 There's no limit to what you can learn or do, unless you choose to stop learning or doing. I'm no motivational speaker, though I've conducted various training courses related to my work, and some outside of my work (such as MATLAB® at a friend's training company). As such, I know the importance of life-long learning and decided to devote this final chapter to discussing some possible areas of expanding your engineering knowledge and skills.

Nothing can substitute practical hands on. Whether Visio is your choice of tool or not, you should at least work through the steps outlined in Chapters 3 and 4 to get a feel of the processes involved in creating a technical illustration, be it mechanical or electrical. There is a 60-day trial version of Microsoft Visio available for download at Microsoft's official website—just Google to find it. Perhaps at the end of the exercise, you may even come to see or agree that Visio is a good diagramming tool, both for its ease of use and functionalities.*

* About half-way into writing Chapter 4, I decided to let a friend take a look at my book and give some feedback. He was so impressed and exclaimed, "I use Visio in my work and never knew you could do THAT with it!"

ADDITIONAL RESOURCES

Sample Visio templates and symbols (layout and schematic) are freely downloadable at www.visio-for-engineers.com. I will also be offering my collection of Visio symbols for component layout packages (discrete, DIP, SIP, ZIP, SOIC, QFP, PLCC, PGA, BGA, etc.) and electrical representations (digital logic, gates, analog, mixed, etc.) for purchase at affordable prices that will save you time and trouble than if you were to create them on your own. Information and prices for these packages can be found at the same website mentioned above once they are made available.

In the course of reverse engineering a PCB, you will no doubt encounter components that are long obsolete or have markings that are difficult to decipher, in which case you may not be able to make out their identities thereby hampering your effort. This is one of the many challenges and frustrations you will face and I can empathize with that. Like you, I've spent countless hours searching for component information and datasheets, and I'll like to help relieve your pain a little if I can.

As a supporter of this book you are welcome to email me (ngkt@engineer.com)[†] and put in your request for unidentifiable components.[‡] I will check if I have the information you need and send it to you. Please allow time for respond as I may be busy with work, going on holidays, feeling blue or attending to some personal matters. Also, I do not guarantee that I'll be able to identify every unknown part—I'm only doing a personal favor on a best effort basis, so don't be upset or disappointed if it did not turn out well. I do get stumped on unknown components myself too. Thanks for your kind understanding!

[†] If you're emailing me the first time, [kindly attach an electronic copy of your receipt as proof of purchase of this book](#). I cannot, and neither have the time to entertain anyone and everyone except those who are willing and kind enough to support my hard work through honest purchase. To these my readers, a big thank you!

[‡] I need to qualify this term to mean components with part numbers or markings that do not register meaningful hits when you search online, just in case I get flooded by emails for parts requests that are readily available online.

One thing to bear in mind—keep your email short and simple. If you're looking for only one unknown part, include its complete part number on the Subject title, then describe simply its packaging (e.g. 16-DIP, 8-SOIC, etc.) in your email. If there are more than one part (kindly limit to a maximum of 3, please!), list them with their packaging in your email. If you're not sure what package it is, you may attach its photo but do keep the file size small without losing too much resolution.

And kindly refrain from asking about any other engineering questions or problems you may have that's unrelated to this book, or that are not within my means or purpose to answer. (I'm serious about this! Really.)

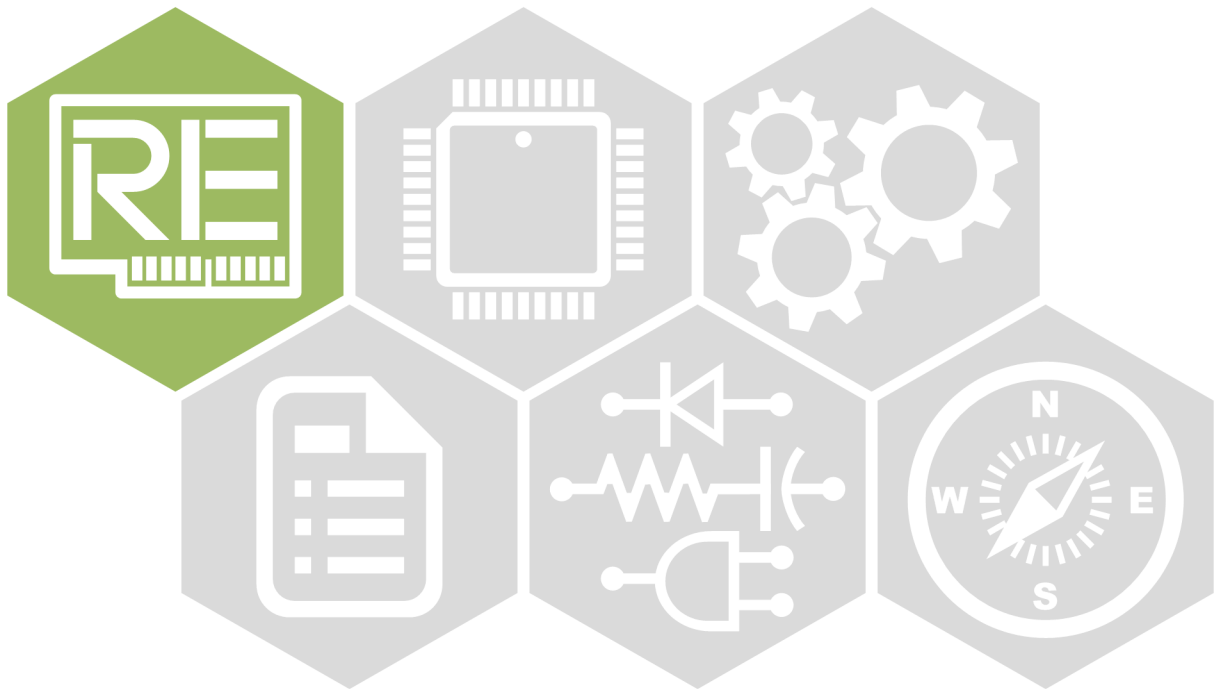
Success is not final, failure is not fatal:
it is the courage to continue that counts.

Winston Churchill (1874-1965)

1

Before You Begin

**Some pre-requisites are essential
to save you from hitting
against those brick walls...**



CONTENT

What You Need to Know-18

Why Bother?-19

My Personal Story-20

Legal and Copyright Issues-25

The Three D's-27

1

BEFORE YOU BEGIN

Printed circuit board (PCB) technology has seen a tremendous jump since its humble beginnings in the early 1900s. From simple discrete, single-sided through-hole to the complex fine-pitch, multi-layered surface-mounted board, the amount and density of components for a given PCB area have increased manifold while the overall size of PCBs have reduced substantially.

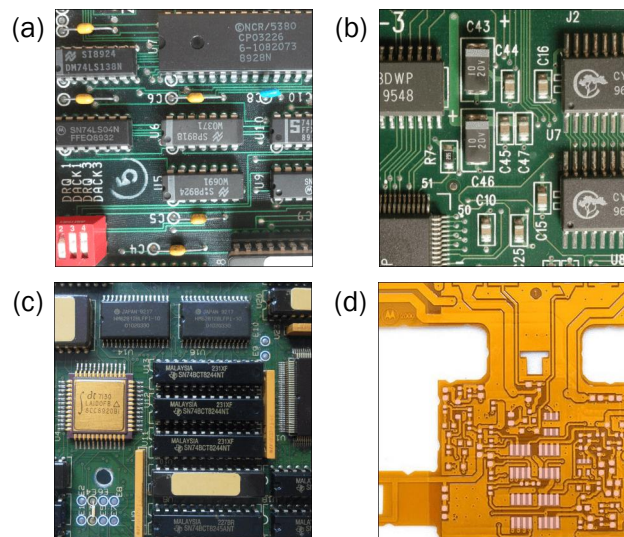


Figure 1.1 Types of PCB: (a) Through-hole, (b) Surface-Mounted (c) Mixed, and (d) Flex-print.

Such rapid advancements in PCB design not only present problems and difficulties to the test engineer who is responsible for writing test programs for the board, they also pose seemingly insurmountable challenges to those daring enough to re-create the schematic drawings. PCB reverse-engineering is indeed not for the uninitiated nor the faint-hearted. But for those who are willing to devote their time and energy to learn this art, the end results can be rewarding to say the least; it may even put you a cut above your fellow engineers because in the process of doing it, you not only unravel the beauty of the PCB itself, you actually gain engineering insights and ideas from the original designer's expertise and ingenuity on how a particular circuit is designed, how certain difficulties are overcome, as well as the sound practices and design techniques that are applied in the real world.

WHAT YOU NEED TO KNOW

While this book is intended for beginners interested to learn PCB reverse engineering, some background in electronics is preferred:

- You should have a working knowledge, or at least be able to read schematic diagrams (that's the reason for buying this book in the first place—to re-create schematic drawings from a PCB!). It will be advantageous if you're already acquainted with the basic rules and concepts of what constitute a good schematic diagram, or better yet, if you have experience with some kind of EDA tools or software. You'll be able to skim through or skip over some of the basic information presented in later chapters.
- You should be able to identify simple discrete devices (resistors, capacitors, diodes, transistors, MOSFETs, etc.), integrated circuit packages (DIP, SIP, PLCC, QFP, PGA, BGA, etc.) and modular components (DC/DC converter, etc.). But not to worry, I will provide useful pointers and tips along the way, when necessary.
- You should have in your possession a good digital multimeter with the diode measurement function on top of the standard functions and know how to use it. I will also introduce other additional tools that will greatly aid you in your reverse engineering effort when we come to the appropriate chapter and topic.
- You should be familiar with the color codes for certain discrete devices (resistors, capacitors, etc.) so you can read their values without having to constantly refer to the reference charts (I've provided quite a collection in the appendices). It's not the end of the road, however, if you're not able to remember what the color codes are; just that it will probably slow you down if there are plenty of such components on the PCB you're working on.
- You should be acquainted with datasheets. Not that you're required to know all the component-related specifications, or understand every aspect of the device's performance. Thankfully no. But you do need to be able to find information like pin number, orientation, signal names, etc. and typical application examples provided by the manufacturer on how the device is used in real-life designs.

At first glance it may sound like quite a handful above, but trust me, it couldn't get any easier than what I have laid out as the basic skills required to get started. Before you even realize it, these would have become second nature and you'll be focusing on the real work!

Then again, there are skeptics who'll say...

WHY BOTHER?

That's a reasonable question. After all, there are machines out there in the market that can do the work more quickly, accurately, and reliably in a week or less, such as those shown below:

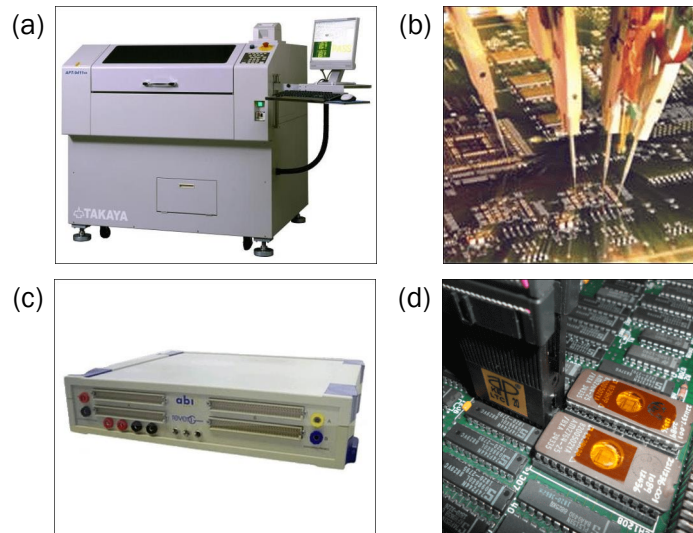


Figure 1.2 (a) Flying probe test system (courtesy of Takaya Corporation), (b) Flying probes in action, (c) RevEng Pizza Box System (courtesy of Abi Electronics), and (d) Test clipping on DIP IC.

So why even bother to learn how to do PCB reverse engineering by hand?

Firstly, flying probe test systems are expensive test equipment. A basic system easily costs over \$100k and that's not including the software license for the reverse engineering option. Even if your company could afford it, nobody would stick out his neck to buy one unless there is strong business justification for the purchase. And we have not yet consider the training and learning curve to familiarize its process and utilization, the yearly calibration and maintenance of the machine, and the cost of ownership just continues to add up.

What about benchtop versions with semi-automated learning of circuit traces? I can think of a number of vendors that supply such a product, namely Abi Electronics, Diagnosys, and Qmax. These products may be cheaper than a flying probe test system, but not necessary affordable, easier to use, or better in delivering quality results.

I've worked with Diagnosys' first generation PinPoint Tester and I can tell you the RE software is a pain to use. You could spend days rearranging and realigning the schematics even if the PCB is moderately complex and it still does not look quite right. While the RE software could export my work out to the standard EDIF format, most of the time I was stuck with its native editor with rather primitive editing features. And the Pinpoint Tester with RE option still costs us about \$50k after discount, even though we were the first users in Singapore!

Chapter 1

As for the simpler and lower cost pizza-box system, a basic configuration with 128 channels will set you back by about \$5-7k at the time of my enquiry. Abi Electronics also sells their more powerful cabinet systems that support from 1024 to 2048 channels to reduce the amount of test clipping on the PCB, but again the cost will invariably go up too. In terms of usage, after defining the components and placement the RE software will guide the operator to clip or probe in clusters, depending on the available channels, and then learns the connectivity of the PCB. The netlist generated is then exported to another EDA tool called EdWin from Visionics, a much more user-friendly and feature-rich schematic editor.

While it sounds simple enough to clip and probe circuit clusters, my experience with Pinpoint taught me the challenge is in ensuring good electrical contact between the test clips and the component pins. This means that the PCB must be stripped clean of any conformal coating, and the test clips must also be free of oxidation and the grip must be good and not weaken as a result of mechanical fatigue from frequent use. If the PCB is surface-mounted or mixed type, you'll need an assortment of SMT test clips to do the job as well, and these QFP and PLCC test clips are by no means cheap!

So the real question is, are you prepared to fork out the money up front for a one-time or ad hoc PCB repair job that requires you to do a bit of reverse engineering? I think the answer is pretty obvious.

MY PERSONAL STORY

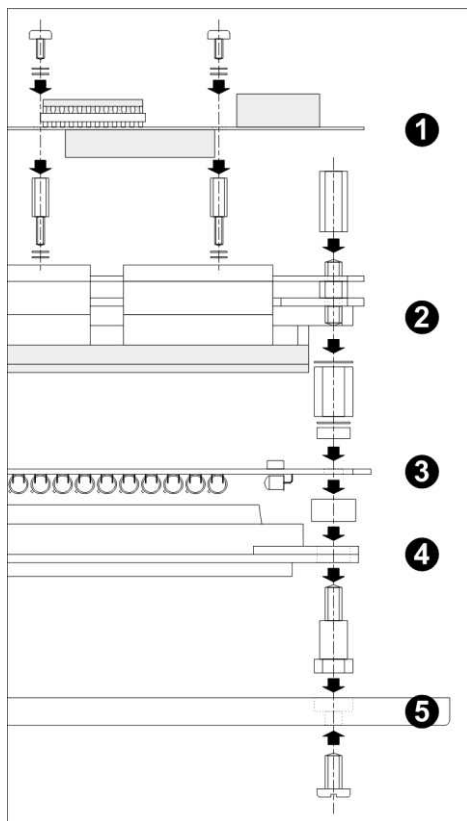
If you bother to read up to this point, I believe that besides commending you for being such a good reader, I should also motivate you to give yourself a chance to try out the approach espoused in this book, a method which I've refined through years of doing reverse engineering myself. And what better ways to inspire you than to share with you some of my own stories, not only as proofs that it's possible but satisfying as well, despite the sweat (figuratively, since I worked in an air-conditioned workshop) and frustrations I've had (sure, if it's easy, everyone would be doing it, and there'll be no need for this book in the first place!).

As a principal engineer working in a defense industry company supporting the military, notably the air force and navy, in the area of local depot-level repair involving weapon systems that are, well, rather dated, it is inevitable that many defective PCBs that are sent to my workshop lack the necessary schematic diagrams to carry out troubleshooting analysis and repair work. Sounds familiar to some, if not many of you who are repair technicians and engineers by profession, isn't it?

My first serious attempt at PCB reverse engineering happened in the spring of 2001, about six months before the fateful 9-11 event. It was a Monday morning when I came in for work and as usual a trip to the pantry to get a cup of hot water. As I passed by the meeting room, I noticed a 16 by 12 square inches by 3-inch high item lying face down at one corner of the conference table. In the course of the day, some of us went in and out of the meeting room, taking curious peeks at the item but no one could figure out what it was doing in our work centre. It was only near lunch time when my manager came back from a meeting that he beckoned me to the room and said, "What do you think—can we repair this unit?" Suddenly it dawned on me that I was tasked to take up the challenge, being one of the senior and more experienced engineers at that time.

After examining more thoroughly I remarked, "It seems to be a high voltage display unit of some sort. We've done PCB repairs all along but this is the first time we're looking at a whole unit." Well, my hunch was right. It's a plasma touchscreen display unit used on-board naval ships. My manager related that the navy had approached him for a solution to their repair predicament—they had been sending these defective units overseas to be repaired by the OEM, and being end-of-life (EOL) items, there's no more contractual support so the cost of repairing each unit was ridiculously high. Lately the frequency of breakdowns had increased due to aging, and the navy desperately needed a solution to cut their overseas repair cost, and fast.

"Let me take this baby apart to study it. I'll get back to you in three days." And so it was that on the third day I reported my findings to my manager.



As shown in the partial cross-section, the touchscreen display unit comprises five sub-parts:

1. CPU control board
2. Display driver high-voltage sub-assembly
3. Infrared matrix sub-assembly¹
4. Touchscreen display frame
5. Front panel mounting

Defect symptom for this unit was no display so I worked on the display driver sub-assembly and found some faulty power MOSFETs. The parts were ordered and replaced but no testing could be carried out since there's no electrical drawing to show connector pinouts and what power to apply. The item was sent back to navy for site test and surprisingly it passed!

I was told that more such units would be coming in from the navy's back logs with quite a number of them having touchscreen with no response failures—an indication that the problem might have something to do with the infrared matrix sub-assembly. It therefore prompted me to consider doing reverse engineering for this particular sub-assembly.

Figure 1.3 Touchscreen display unit cross-section.

The next consideration was what CAD software to use for this endeavor. It so happened that there's an old copy of Visio® Technical 4.5 lying around, so I installed and fired it up to do some fiddling around. It made an instant connection that persuaded me this was the best tool for the job! I will elaborate more when we discuss using Visio later.

¹ The names for all the five sub-parts were coined by me, but I particularly like the infrared matrix sub-assembly because it was inspired by the blockbuster movie The Matrix (1999), and for the fact that it has a 40 x 30 matrix of infrared emitter and sensor pairs.

Chapter 1

The first thing I did was produce a mechanical drawing² of the front panel, as shown below, which took me just slightly over half an hour. Not bad for a start, eh?

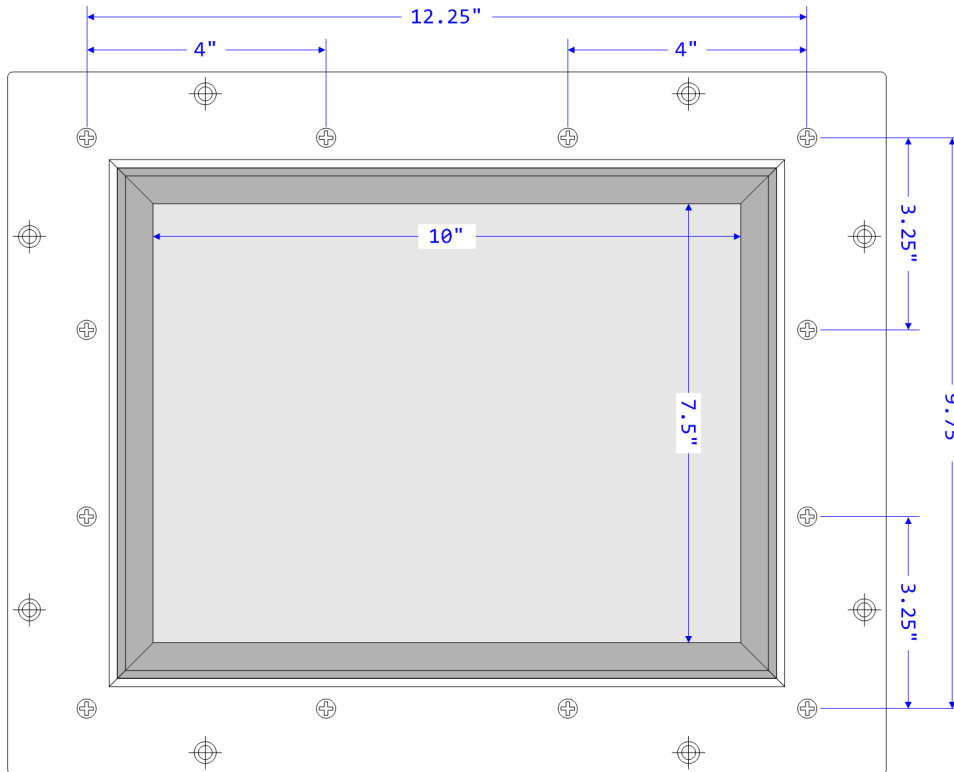


Figure 1.4 Front panel layout drawing of the plasma display unit

The rear view of the unit was more complicated with the CPU board and the display driver sub-assembly visibly in view, but I managed to draw that too though at a much later date when I had the time to do so. However, I will not be showing it here since that is not the main focus of the story. Still, I'm sure you'll agree that Visio fared pretty well for illustration purposes (and we're just warming up!).

After disassembling the unit, I proceeded to analyze the infrared matrix sub-assembly and gathered whatever information I could find on its components. What intrigued me was the primitive yet beautiful design of this multi-layered PCB—the making use of narrow-angled beam infrared emitter and sensor diode pairs—to achieve the X and Y coordinates mapping for the touchscreen effect that was the usual practice back in the mid-1990 era. It had many of the elements that made reverse engineering particularly interesting and challenging, such as the peculiar shape and size of the PCB, the presence of through-hole and surface-mount components that are mounted on both sides, and the thick conformal coating. That's quite a handful for my virgin attempt but I went ahead anyway.

² Incidentally, one of my favorite subjects during my higher secondary school days was technical drawing. Back in 1978-79, we did not have personal computers. Instead, we used a hand carry drawing board with a T-square, two square rules (30/60° and 45°) and mechanical pencils of 0.5 and 0.7 lead diameters to draw on A3 size papers!

You can see from the photo that the infrared matrix sub-assembly is quite a big board with a rectangular cutout in the middle for an array of horizontal (40) and vertical (30) pairs of infrared emitter and sensor diodes. These fit nicely around the tinted bezel display frame in which the brightly orange-lit plasma display of the display driver sub-assembly shows through (see Figure 1.3).

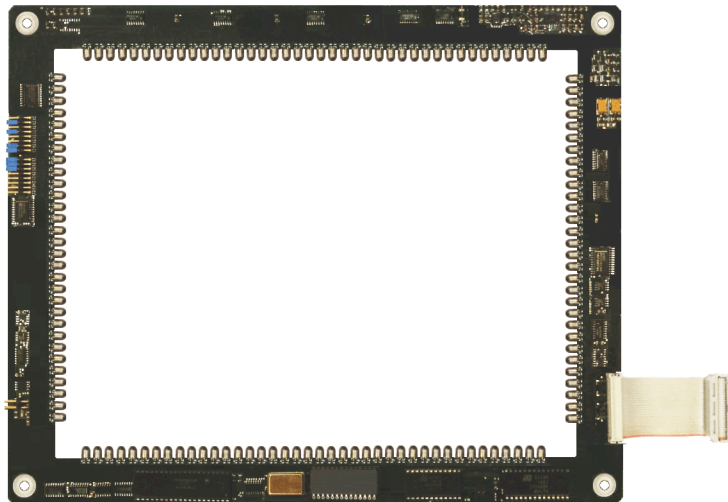


Figure 1.5 The infrared matrix sub-assembly (top view)

The layout drawing of the infrared matrix sub-assembly is shown overleaf. There are two sheets to the original drawing I made but I'm showing just the first sheet, which includes the parts list of all the IC components found on this side of the PCB, as well as magnified sections on some areas of the PCB that are too small to display the reference designations of SMT discrete devices. This is one method of illustration that will be touched on when we come to [Chapter 3](#).

If you work often on surface-mounted or mixed PCBs, you'll notice that apart from the ICs and bigger component parts, the miniature SMT resistors, capacitors, and even semi-conductors such as the SOT package³ diodes and transistors may not have reference designators assigned to them on the silkscreen layer, and this is expected with densely populated SMT boards due to the difficulty and impracticality of doing so. We will explore how to give these nameless discrete devices their reference designators—to facilitate the drawing of the schematic diagrams—in the same chapter mentioned above.

To cut the story short, in total I spent about a month drafting the parts list, gathering the datasheets, creating my first collection of Visio symbols for both the layout and schematic entities, drawing the PCB layout, and finally tracing out the schematic diagram.⁴

³ SOT stands for Small Outline Transistor and they come in many form and sizes, most commonly SOT-23, SC-70, and SOT-223. Some diodes make use of such packages, utilizing only two of the three terminals (e.g. BAS16 as shown in Figure 1.6).

⁴ As much as I would like to, the schematic diagram for the infrared matrix sub-assembly will not be shown due to inherent copyright issues which are discussed in the next section. Suffice it to mention that the drawings spanned four A3-size pages.

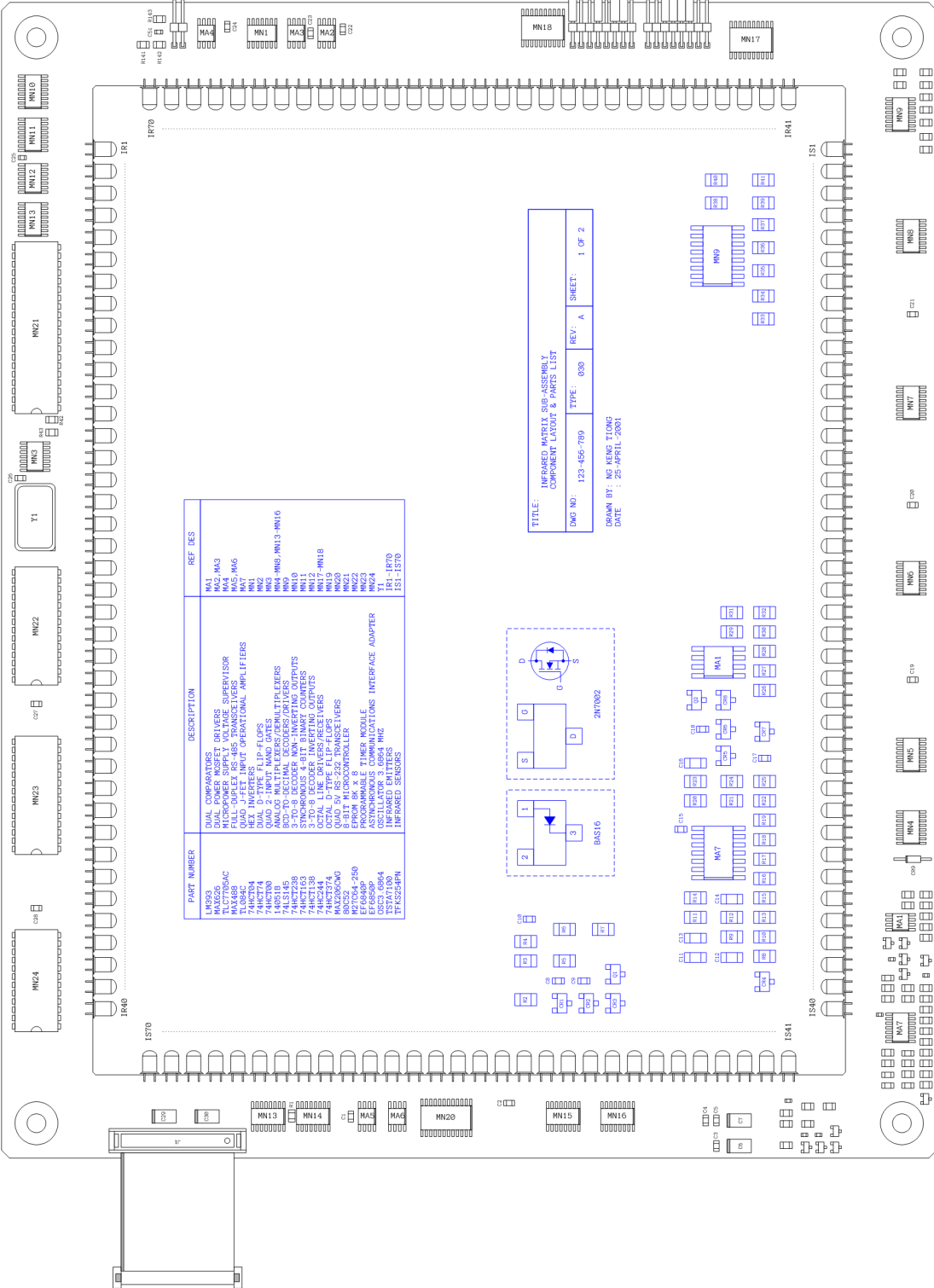


Figure 1.6 Layout drawing of the infrared matrix sub-assembly

LEGAL AND COPYRIGHT ISSUES

Probably one question you have in mind when approaching the subject of PCB reverse engineering is copyright issues. When I started out to do PCB reverse engineering, this question did not cross my mind because as an engineer, it has always been my nature to be curious as to what makes a PCB tick, and to look for solutions to problems or challenges I encounter at work. It was only when my company started patenting its own products, instilling and reinforcing the awareness of intellectual property rights in its staff, that I realized the need to check if I had in anyway infringed on copyright laws.

So to what extent is PCB reverse engineering legal, and under what circumstances is it a violation of the copyright law? Given the complexities of legal terms involved, there are simply no straight answers.⁵ Still, the fact is there are companies out there openly advertising and providing PCB reverse engineering services,⁶ as well as selling software tools that perform such tasks. To my personal understanding, such services exist for some of the following reasons:

- The PCB is obsolete and no longer supported by the original equipment manufacturer (OEM) or in some cases, the OEM is out of business and supply of the PCB is unavailable in the market. In such instances, the only recourse then is to engage these companies to reverse engineer and re-produce the PCB in sufficient quantity to extend the useful life of the system.
- The company or OEM that designed and produced the PCB lost their design files (whether by accident or through catastrophic causes⁷) and are still under contractual obligation to support their product before its end-of-life (EOL) period.
- Modification to the original PCB design is required for extended capabilities but due to nature of confidentiality, the OEM cannot be involved or have foreknowledge of such changes, in which case obtaining the OEM's design files will be out of the question.

I have visited a company that has been in this business for many years. The way they go about doing PCB reverse engineering is first to obtain a bare board from their customer, then proceed to carefully and meticulously remove layer by layer to reveal the artwork, and using a very high resolution scanner to scan each layer for correlation of the tracks and vias, before converting the data into gerber format for reproducing the PCB, without having to re-create the schematics. This approach requires the customer to willingly sacrifice a good PCB, or at least one without physical damage, though not necessarily working. Primitive but effective in addressing the needs of the first kind of customers mentioned above.

⁵ For those interested to know more about copyright issues pertaining to this subject, a good article can be found in the paper written by David C. Musker at <http://www.jenkins.eu/articles-general/reverse-engineering.asp> which he presented at "Protecting & Exploiting Intellectual Property in Electronics", IBC Conferences, on June 10, 1998.

⁶ These companies usually spelt out clearly in their websites the contractual terms by which they carry out their trade, including the level of confidentiality and non-disclosure agreement that must be respected by both parties—the customer and the service provider.

⁷ Now don't laugh—it does happen because I know of one company that had their database servers crashed due to power outages and their valuable design data all wiped out or corrupted.

Companies engaging in such services, however, are able to provide CAD data (netlists, schematics, bill of materials, gerber files, etc.) as a complete package. I would suspect that they at least own a flying probe machine to carry out the tedious and monotonous continuity measurement, after they digitized the PCB and obtained its X-Y coordinates data for probing references. This solution will certainly meet the needs of the other two kinds of customers.

So what can we conclude from the scenarios presented? Whilst copyright laws exist to discourage or keep at bay would-be competitors (or even pirates) from stealing the designs and ideas for their own commercial gains,⁸ PCB reverse engineering services are still very much sought after, more as a desperate attempt to resolve the genuine problems of obsolescence or lack of support, and in such cases the PCBs reproduced are usually intended for internal consumption and not external circulation. Confidentiality and non-disclosure agreement (NDA) will also ensure that only the customer and the service provider are the only parties in the know.⁹

It's a fine line we're treading when we do reverse-engineering of any sort. We need to ask ourselves why we're doing what we're doing, and whether we're doing it out of necessity on a personal capacity to get our job done, or out of curiosity when a certain PCB design so intrigued and interests our engineering senses that we couldn't pass it by without knowing what makes it tick. Frankly, troubleshooting and diagnosing PCBs without documentation invariably involves some degree of reverse-engineering, albeit in a less systematic and perhaps haphazard way. Prior to taking up reverse-engineering using Visio, I did what many repair technicians or engineers would do when stumped with a PCB without schematics—create partial sketches of the circuits I was analyzing, hand-drawn or with the help of primitive graphics editor.

Again, I need to reiterate that this book will not teach you how to use EDA tools to enable you to reproduce the PCB that you want to reverse engineer. Rather, I am offering you my experience as an engineer who faces the same challenge you may be facing—lack of information to carry out effective repair on a PCB—and how you can overcome it by learning to re-create the schematic drawing, in part or full, to enable you to get your job done.

Before we start our journey, here are some last things you need to know...

⁸ Some OEMs go so far as to use programmable logic devices or embedded designs with security bit protection, or even produce their own ASIC or proprietary components, in-house or out-sourced, just to be sure that there is no way to reproduce a functional copy of their product but to go back to them for repair. But with the increasing cost of manufacturing and shorter time-to-market product cycle to stay competitive, as well as customers not willing to be locked in by a single manufacturer—if they can help it, commercial-off-the-shelf (COTS) parts are gaining wide acceptance among system integrators like my company and military organizations facing obsolescence issues and discontinued support due to longer operational life requirement which OEMs are increasingly unable to meet.

⁹ Having said that, I personally do not endorse PCB reverse engineering to be used as a tool for anti-competition or theft of design, in which the end result is the financial lost of the rightful designer or company concerned. I believe that as an engineer, there is a code of honor to live by and uphold, by which personal enrichment and knowledge can be gained and shared, but not at the expense of others.

THE THREE D's

PCB reverse engineering is by no means an easy task. Very often, it is laborious because of the repetitive nature in finding and ascertaining the electrical connections between components. Also, being able to constantly visualize the schematic diagram that is slowly taking shape and having to realign the various portions of the circuitries to ensure readability requires certain artistic affinity—a skill that can usually be picked up and refined only with practice.

Ultimately, it comes down to three essential traits that a person who wants to engage in such a work must possess in order to be able to successfully complete the task:

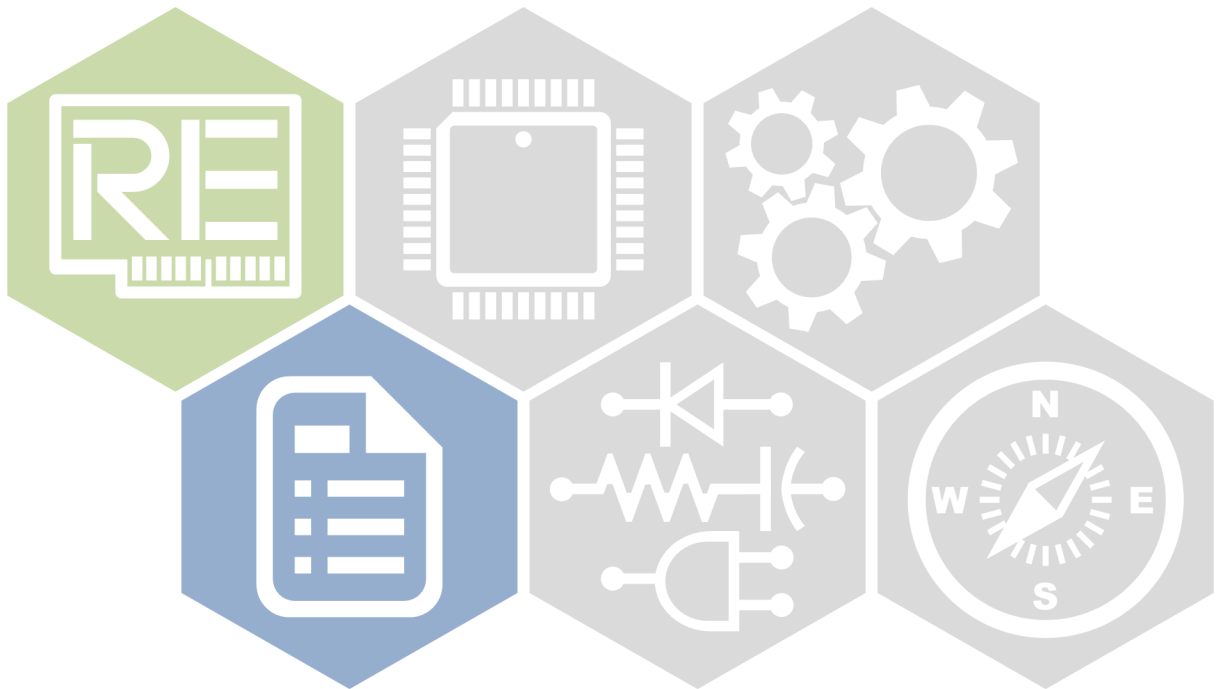
- **Determination.** A 'never say die' attitude is important especially when faced with the constant frustration of not being able to locate the electrical connections to make sense out of the partial schematic diagram. This is not to say that one must be insistent on finding the 'missing link' and gets stuck on one particular part of the PCB, and not able to make further progress. Working smart is just as important as working hard!
- **Discipline.** Being focused, systematic and maintaining a careful work practice are necessary to avoid unwanted mistakes that can prove to be both time-wasting and confusing, perhaps even to the point it demoralizes you. Many engineers have the habit of diving straight into a project or task without giving thought to at least some planning or preparation work. This is a BIG mistake that will pay grave dividends—believe me—I made them myself and thankfully survived to regret (and learn from) it!
- **Dedication.** Nothing can replace the satisfaction of seeing a task complete if a person does not take ownership of the work and prides himself or herself in the thing he or she is doing. After all the end product bears the signature of the craftsman!

Enough said—let the fun begin!

2

Basic Preparation Work

**Good preparation upfront
will give you a good head-start
and a well-deserved finish later!**



CONTENT

Getting to Know Your PCB–31

Accessibility–32

Bill of Materials–33

Resistors and Networks–35

Capacitors–37

Inductors and Transformers–40

Fuses–42

Relays and Switches–43

Crystals and Oscillators–45

Diodes and Zeners–46

Transistors and MOSFETs–48

Integrated Circuits–49

Digital Logic–50 Memories–50 Programmable Logic–51 Processors (CPU)–52

Peripherals and Chipsets–52 Analog (Linear)–53 Hybrids–53

Military Part Numbers–53

Miscellaneous–54

Delay Lines–54 DC-DC Converters–54

Components without Markings–55

Missing Reference Designators–56

Filling up the BOM–58

Conformal Coating–60

Personal Protection Equipment (PPE)–61

Component Datasheets–62

2

BASIC PREPARATION WORK

"If you know both yourself and your enemy, you can win numerous (literally, a hundred) battles without jeopardy."

Sun Tze, The Art of War, c. 544-496 BC

The same is true when it comes to reverse engineering a PCB. I have mentioned in the previous chapter the background that is expected of you—that's knowing your abilities and the areas you may be lacking for a start. Not that you can't start right away. It'll be good though to make an inventory list of the areas you need to brush up and get back up to speed, and where you really need to put in effort to build up that knowledge.

As for knowing your enemy—well, that's what this chapter is all about.

GETTING TO KNOW YOUR PCB

The printed circuit board is a marvelous invention, combining many disciplines (material, mechanical, chemical, electrical, thermal, etc.) into one coherent entity. Indeed it is inconceivable what the world would be like without this entity called PCB, since our modern lifestyle depends heavily on it to run our daily lives, from household items (TV, hi-fi system, fridge, washer, aircon, handphone, etc.) to industrial infrastructures (transportation, communication, entertainment, etc.), you name it—it's everywhere!

I don't know about you, but as an electronics engineer, being able to look at a PCB and appreciate the ingenuity of the designer that brought it into existence,¹⁰ from initial ideas to implementation, from its baseline specifications to its intended functions, and from the choice of components to its final layout to achieve the best possible use of limited board space, it's more than just engineering and planning. There is an artistic beauty that is visible only to the initiated, and a hidden gem that is revealed to the keen-eyed. In every PCB there is a treasure to be discovered, some coveted design knowledge to be gained; and that is a strong enough reason and motivation for the hard work ahead.

¹⁰ Before the advent of electronic simulation software, and when components were not as complicated as today's VLSI technologies, design engineers built prototypes by breadboarding or wire-wrapping on vero boards for proof of concepts.

The following are the key areas you need to pay attention to and gather as much information as you can before launching into the deep:

- Accessibility (probe points)
- Bill of materials
- Conformal coating
- Datasheets

I call these the ABCD's of preparation work. By working through these key areas, you'll gain a better overall picture of a PCB, which may help you decide if it's feasible to attempt, if you have the time and ability to do it, and whether it's worth the effort or not.

ACCESSIBILITY

Generally, PCBs can be grouped into four types: through-hole, surface-mounted, mixed, and flex-print. Prior to the appearance of ball-grid-array (BGA) components, majority of the PCBs have little or no accessibility issues since the component leads or pins are visibly soldered to the board by machine or by hand. There are exceptions, however, when a manufacturer decides to apply some kind of epoxy compound over certain parts of the PCB to provide support for weak spots or socketed components, or simply to mask off sensitive information to keep prying eyes away or discourage repair and rework.



Figure 2.1 PCB with BGA ICs (courtesy of PicoChip¹¹)

SMT technology has, however, evolved to the point where even more precious board space savings can be achieved on the PCB, by transforming the through-hole pin-grid-array (PGA) a step further to the surface-mount BGA. Such progress is seen as inevitable and necessary in engineering expediency and business sense, but not necessarily appreciated and welcomed by the test and repair communities, since it severely hampered the ability to test and repair this type of PCBs using traditional methods and equipment.¹²

¹¹ PicoChip was a venture-backed fabless semiconductor company based in Bath, England, founded in 2000. In January 2012 it was acquired by Mindspeed Technologies, Inc. (Source: Wikipedia)

¹² The often hyped alternative solution to this predicament is boundary-scan or JTAG testing. Granted, many new ICs including the BGA types have in-built JTAG circuitries, but ensuring that a PCB meets the JTAG requirement will take additional efforts on the designer's part to implement to realize the design-for-testability (DFT) goal.

Accessibility of probe points is thus an important factor to consider before reverse engineering work can commence. You should therefore make an estimate of how much of the PCB's component leads or pins are accessible, compared to those that have been obstructed, hidden, or even buried intentionally. One other factor that adds to the problem of accessibility is conformal coating; we will look at it separately since it's quite a broad subject in its own right.

Meanwhile, ask yourself the following questions:

1. Do I intend to reverse engineer the whole PCB or just a portion of it? If it's the latter, will there be accessibility issues to contend with?
2. For PCBs with obstructions or BGA components, do I choose to remove them in order to access the hidden probe points, or do I define boundaries and work by clusters?
3. Is the conformal coating, if present, removable for ease of probing?

If you can get past this first hurdle, then it's time to proceed to the next step.

BILL OF MATERIALS

The components that populate a PCB vary widely in types as well as quantities. The purely analog PCBs usually have more discrete passive and semiconductor devices than ICs, whilst digital PCBs will contain mostly ICs and lots of decoupling capacitors plus a handful of pull-up or terminating resistors and networks thrown in.

The bill of materials (BOM) associated with a PCB provides a complete listing of the components present on that board, as well as possible optional ones depending on the PCB revision and configuration. It would be great to have the BOM readily available as it definitely cuts down the amount of work to create one yourself, and eliminates guess work and further effort to determine the values of components such as resistors, capacitors, and other surface-mounted devices (SMDs) which are too small to be marked.

On top of that, most military-grade PCBs have one peculiarity—the ICs used are of military specification (MIL-spec) type that do not exhibit the familiar standard industrial part numbers (e.g. SN74LS00), but have cryptic and hard to decipher NATO or national stock numbers (NSN),¹³ or MIL-STD-883¹⁴ part numbers (e.g. M38510/30001).

¹³ The NSN is a 13-digit numeric code, identifying all the 'standardized material items of supply' as they have been recognized by all NATO countries including the United States Department of Defense. (Source: Wikipedia)

¹⁴ Components that achieved the MIL-STD-883 grade are suitable for use within military and aerospace electronic systems. It has been the stringent requirement imposed for all military grade PCBs in the past, but due to shortage of supply and obsolescence arising from acquisitions and folding of semiconductor companies, the US military has loosen the requirement to include industrial grade components as alternate replacement, if necessary.

To create a BOM for the PCB, you need not go into details the way the manufacturers do for their products. It's meant to be for your reference so keep it simple with just the two essentials—part number and reference designation. Below is a sample BOM which I made from a military PCB:

Table 2.1 Sample Bill of Materials

PART NUMBER	REFERENCE	PART NUMBER	REFERENCE
54F00	U47	RNC55H16R2FSCJ	R100
54HC221	U40-U41	RNC55H51R1FSCJ	R96-R97
54HCT244	U38-U39	RNC55H68R1FSCJ	R102-R108
54FCT245	U45-U46	RNC55H75R0FSCJ	R93
54HCT245	U36-U37	RNC55H82R5FSCJ	R109
54FCT373	U42-U44	RNC55H90R9FSCJ	R78
54HCT373	U34-U35	RNC55H1100FSCJ	R80-R82
71256	U7-U10	RNC55H1470FSCJ	R79
		RNC55H1630FSCJ	R77
AD7524SQ/883B	U30	RNC55H2150FSCJ	R73-R74
AM26LS31DMB	U51	RNC55H2160FSCJ	R86
AM26LS32DMB	U50	RNC55H2370FSCJ	R76, R83
CLC430	U25-U27	RNC55H2870FSCJ	R84-R85
HA2-5002/883	U22-U24	RNC55H6810FSCJ	R72
HA7-2544/883	U21	RNC55H9090FSCJ	R70-R71
HI1-201HS/883	U19-U20	RNC55H1101FSCJ	R23-R42
LF198H/883C	U17	RNC55H2151FSCJ	R59, R61
LM117H/883B	U16	RNC55H2371FSCJ	R57
LM119H/883QS	U14	RNC55J4641BSCJ	R43-R53, R55-R56
M55310/16-B41A	U52 (40M00000)	RNC55H5111FSCJ	R63-R65, R68
		RNC55H9091FSCJ	R58
M82C288-10	U31	RNC55H1052FSCJ	R8, R11-R13
MD82C284-12/883	U32	RNC55H1582FSCJ	R20
MG80C286-12/883	U33	RNC55H1003FSCJ	R4-R6
MG82786/B	U12	RNC55H1633FSCJ	R2
MP7684SD	U11	RNC55H3163FSCJ	R3
MT5C2568C-25	U5-U6		
REF01J/883	U2	M8340109M1002GC	Z1-Z3 (10K)
JX4454-1	CR4-CR6, CR8-CR9, CR11-CR15		
JX5712-1	CR10		
JX1N5907	CR1-CR3		
M39003/01-2729J	C2-C3	47UF	
M39003/03-0323J	C1	68UF	
M39003/03-0318J	C4-C10	8.2UF	
CCR05CJ4R7DRV	C108-C109	4.7PF	
M39014/01-1296	C112	68PF	
M39014/01-1317	C106-C107	1NF	
M39014/01-1541	C104	22NF	
M39014/01-1547	C105	47NF	
M39014/01-1553	C86-C99, C101-C103, C107	100NF	
M39014/01-1581	C110-C111	22NF	
M39014/02-1320	C56	1.5NF	
M39014/02-1415	C12-C48, C55	1UF	

Notice the MIL-STD-883 part numbers for most of the ICs ending in /833 or /833B? Even the discrete components are MIL-spec grade! The values of the resistors can actually be determined from their part numbers, but the capacitors will need referencing to the manufacturer's look-up tables based on their groupings (i.e. M39003/01, /03 and M39014/01, /02).

Let's now look at identifying some of the common types of components.¹⁵

¹⁵ We'll not be able to cover all the possible types of components, or even each type in great details since that will make this book a very large volume. The pages that follow are meant to give you an overview and feel of what you can expect in the vast universe of electronics.

RESISTORS AND NETWORKS

The resistor is one of the most basic element in an electrical circuit. Practically all PCBs will contain these passive components, numbering from a few to over a hundred; and they come in different made and shapes, values and wattage ratings.

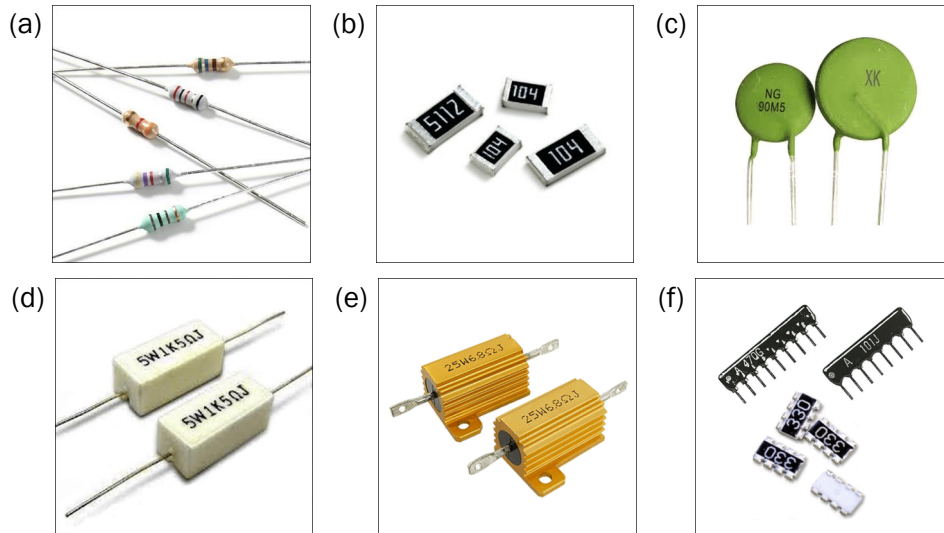
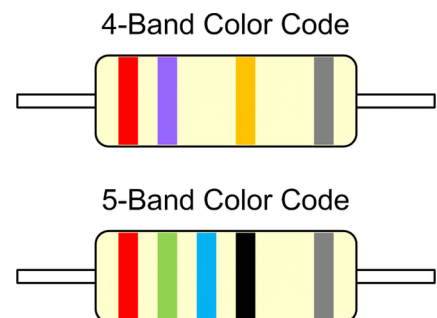


Figure 2.2 Types of resistors: (a) Carbon composite , (b) Chip SMD, (c) PTC thermistor, (d) Cemented wirewound, (e) Aluminum case, and (f) Network.

I remember doing my first project in the second year of my Polytechnic study and was required to design and build a linear adjustable power supply. That was the first time I visited Sim Lim Tower where you could get most of your electronic parts from those family-owned hobbyist shops. The resistors I used back then were the simple carbon composite axial lead type (2.2a) in which the values were marked on the body using color stripes. I had no problem identifying the resistor values since I had committed the color codes to memory. Not sure if this is still a requirement for electronics students today, though.¹⁶

These axial lead resistors usually have 4, 5 or 6 color bands, though the first two are more commonly found. The first few colors that are closely banded together provide the resistance values, while the last isolated band denotes tolerance, and in the case of the rare 6-color band resistor, the temperature coefficient. No idea or can't recall the color codes? No worries! Go over to [Appendix A-1](#) and you'll find a handy color code chart.



¹⁶ Some years ago I was doing troubleshooting work with a colleague in a military unit, and a young technician who was assigned to us was so impressed when I read out the value of a color-coded resistor. I was like, "Huh? What's the big deal?"



Other types of axial lead resistors include the wirewound and metal film. Some of these have their values embedded within their part numbers and printed in multiple lines over their slim profile bodies. The right shows a metal film resistor inscribed with letters and numbers RLR7C4752F and some other alpha-numeric on the back. The numbers 4752 denotes a value of 47.5k ohms (i.e. the number 2 denotes two 0's behind the first 3 digits, so 4752 = 47500 or 47.5k).

SMD resistors are also called chip resistors and they come in many sizes, but notably sizes 0603, 0805, 1206 and 1210 are the more commonly seen.¹⁷ The left column is 3-digit codes and the right column is 4-digit codes,¹⁸ so 223 is 22000 or 22k, and 8202 is 82000 or 82k. The middle row has what is known as radix point codes, that is, the R in between the numbers represent a decimal point, so 4R7 is 4.7 and 0R22 is 0.22, etc. The last row is simply a zero ohm value resistor in both cases.

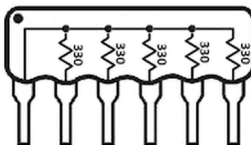


Obviously, SMD resistors are not limited to just chip resistors per se. Some take on similar package like the SMD diodes known as MELF (metal electrode leadless face), and within this category there's the micro-, mini-, and standard MELF sizes. Feeling overwhelmed already? Don't be. We're just getting warm-up... Anyway, whatever forms or packages these two-terminal resistors may appear on a PCB, you'll learn to identify them in no time with practice, even if there are no markings on them!

Resistor networks (also known as arrays or packs) are an interesting collection of resistors arranged in a variety of configurations or patterns within a package, usually single inline (SIP) but can also be dual inline (DIP) just like an IC. The simplest SIP configuration is a group of resistors tied at one end to a common point or pin, usually pin 1 and the other end to individual pins of the package. So a 6-pin SIP will have 5 resistor elements with one common point at pin 1.



The simplest DIP configuration is a collection of resistors lining up freely with their terminals at adjacent pins of the package. Thus, a 14-pin DIP will have 7 free resistor elements.



The simple SIP configuration is normally used as pull-ups or pull-downs for open-collector or termination circuits, while the simple DIP configuration, especially the SMD type, is found in PCBs where board space is a constrain and resistors of similar values are used in abundance.

¹⁷ These numbers are imperial codes that define the sizes of rectangular SMD two-terminal components (mostly resistors and capacitors). You can find a detail comparison of imperial and metric sizes in [Appendix A-2](#).

¹⁸ There is another type of code marking known as the EIA-96 marking method primarily for precise SMD resistors of 1% tolerance and particularly the 0603 size. See [Appendix A-5](#) for reference.

Complex SIP and DIP configurations exist though they are not commonly used except in highly analog circuits. One example is the ladder network used in D/A conversion. Manufacturers such as Vishay, Dale, and Bourns make specialized resistor networks based on bulk orders, and they do carry limited selections of the more popular ones. There is a chart in [Appendix A-3](#) that gives some examples of resistor network configurations you are likely to encounter in the course of your work.

There is a reason why I elaborated a little on these resistor network configurations—they pose potential problems when you start to work on PCBs that have them, one of which is interference with verification of circuit topology. Another challenge you'll face is small SMD resistors¹⁹ with no values on them. See section [Components Without Markings](#).

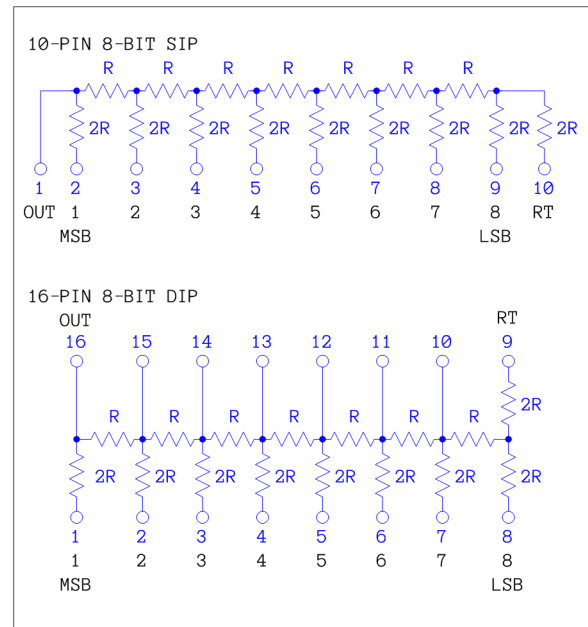


Figure 2.3 Common node SIP and DIP resistor networks.

CAPACITORS

Next to resistors, capacitors are one of the most common components you'll find on PCBs. And like their cousins, capacitors come in different made and shapes, values and voltage ratings as well.

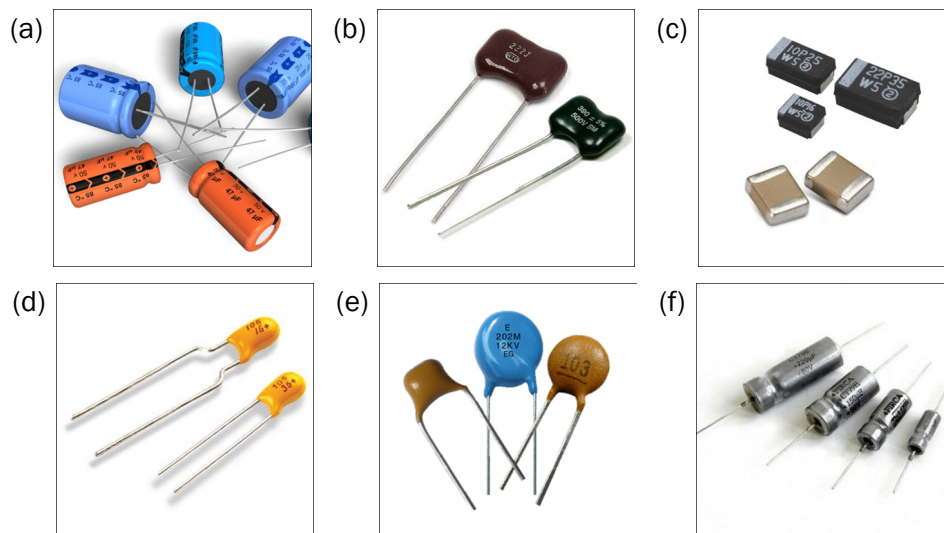


Figure 2.4 Types of capacitors: (a) Electrolytic, (b) Mica, (c) Chip SMD, (d) Dipped tantalum, (e) Radial-lead ceramic, and (f) Axial-lead tantalum.

¹⁹ The same can be said of discrete SMD components such as the capacitors and diodes, and these are more likely to have minimal or no markings on them than their resistor counterparts.

Leaded capacitors come in two flavors, axial and radial leads, the latter being more commonly seen. Unlike resistors, capacitors can have polarities which, if connected wrongly will result in devastating consequences. The varieties of the capacitor can rival the resistor, if not more.²⁰ Besides classifying them as polarized and non-polarized, capacitors are broadly grouped into four major types: ceramic, film, electrolytic, and super. Of course, there are capacitors of other dielectric medium (vacuum, air, mica, glass, etc.) but these are less common and will not be discussed.

Older, vintage capacitors also have color markings just like some resistors as shown in the color code chart in [Appendix A-1](#). One example is the metallized film capacitors made by Philips (right). It has 5 color bands, of which the first 3 bands, starting from the top, represent two digit numbers (15) and a multiplier (10^4) translates to 150000pF or 0.15uF. The white band denotes a tolerance of $\pm 1\text{pF}$, and the last red band is the voltage rating of 250Vdc.



Like the radial-lead ceramic disc capacitor, many capacitors have their values printed in 3-digit form (left). The numbers are similar in meaning to their color counterparts, in this case, $104 = 10 \times 10^4 = 100000\text{pF}$ or $0.1\mu\text{F}$. Sometimes, a smaller value capacitor such as those in the pico-farad range will simply have two numbers only, so a 22pF capacitor will just have the number 22 on it. Radial-lead ceramic disc capacitors were often used as decoupling capacitors in older PCBs, but with improved processes these are being replaced by the better performing multilayered ceramic capacitors.

Film capacitors are known by their many dielectric variations, such as polypropylene (PP), polyester (PET), polyethylene naphthalate (PEN), polyphenylene sulfide (PPS), polytetra-fluoroethylene (PTFE), polystyrene (PS), polycarbonate (PC), and paper or mixed (MP) film.²¹ Now that's quite a mouthful to chew! Fortunately we're not going to commit all these names to memory. It's good to know they exist, though.



Electrolytic capacitors are polarized in that they have positive and negative terminals, axial or radial. They also have much larger farad values, enabling them to store more charges. There are two types of dielectric medium for electrolytic capacitors: aluminum and tantalum,²² and they can be non-solid or solid. These are usually used as filtering capacitors in power supplies and inter-stages of audio amplifier circuits. High-voltage electrolytic capacitors have either snap-in or screw terminals instead of leads, and it is not unusual to find values above $40000\mu\text{F}$!

²⁰ Manufacturers like Kemet, AVX, Vishay, Nichicon and Murata offer a wide selection of capacitor types. It is not possible to cover every type within the scope of this book. You can visit their websites and download their product catalogues for more information.

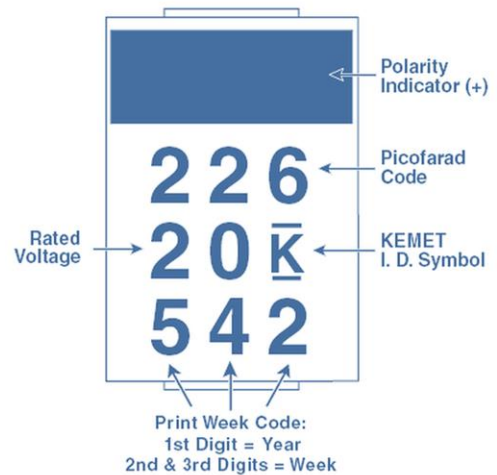
²¹ Source: Wikipedia, under film capacitors. These are also known as metallized film capacitors.

²² There is a third type: niobium oxide, but the electronics industry has been slow to adopt it despite being a low-cost alternative to tantalum, due to technical issues such as lower voltage range, higher DC leakage and ESR compared to tantalum capacitors.

Solid tantalum SMD capacitors such as that shown on the right usually have their values and ratings printed, so it's no problem reading the data. The illustration is self-explanatory (courtesy of Kemet).



Increasingly solid aluminum electrolytic capacitors are finding their ways into PC motherboards. Besides the SMD advantages, they do not have the problem their non-solid siblings has—the tendency to leak and cause corrosion to their surrounding areas after some years of operation.



There is a class of military grade ceramic capacitors known as SMPS capacitors that are used in high-speed switch mode power supplies for their extremely low ESR/ESL compared to electrolytic or tantalum capacitors. These capacitors are also stackable (see figure) to achieve higher capacitance storage capacities. I've personally seen quite a few of these used, one at the end stage of a voltage multiplier to stabilize and sustain a -85V reference output for grid voltage generation purpose; another grouped in banks of three in a launcher unit.

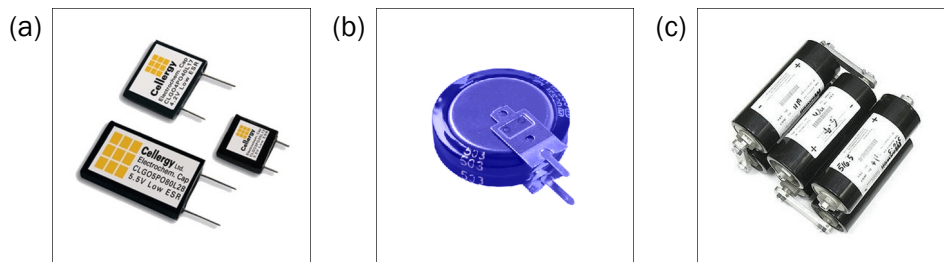
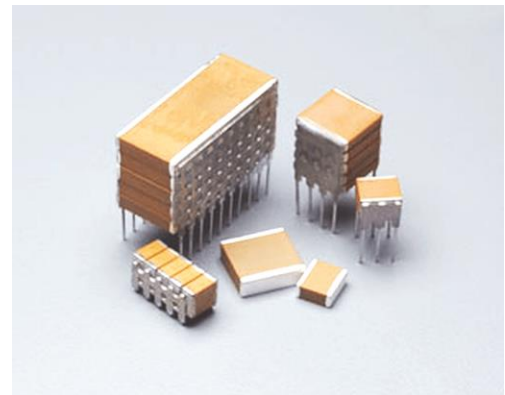


Figure 2.5 Types of super-capacitors.

The last type of capacitors we want to talk about are the super-capacitors. You can think of it as a cross-breed between a conventional capacitor and a rechargeable battery (a kind of hybrid), and yes, it is polarized. While I have yet to encounter one in my work, looking at the broad spectrum of modern applications²³ these capacitors are used, it won't be long before we come across them.

²³ Low supply current for memory backup in SRAMs, power for cars, buses, trains, cranes and elevators, including energy recovery from braking, short-term energy storage and burst-mode power delivery, etc. are some examples. (Source: Wikipedia, under super-capacitors)

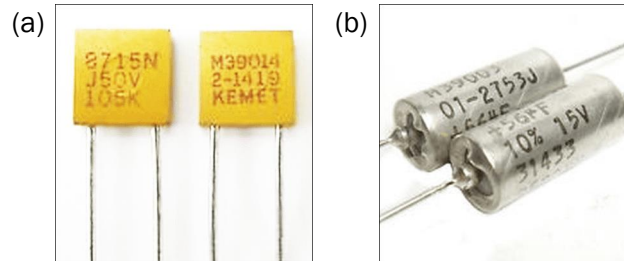


Figure 2.6 Military grade capacitors: (a) M39014, (b) M39003.

Military grade capacitors employ different markings that differentiate them from the commercial and industrial grade ones. Most common among the ceramic types are the M39014, and for the axial lead tantalum types, the M39003. Usually you should be able to find the values or standard notations either on (a) the opposite side, or (b) below and around the military part numbers, as shown above. You can refer the datasheet or catalogue interpretation for these notations, by checking the last 4-digits against a look-up table to determine the farad value and voltage rating.

INDUCTORS AND TRANSFORMERS

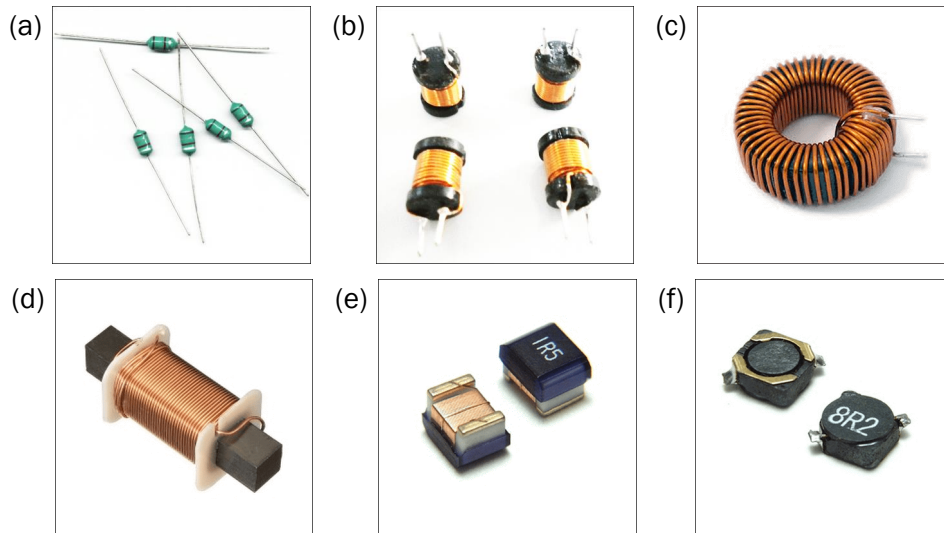


Figure 2.7 Types of inductors: (a) Axial-lead, (b) Radial-lead, (c) Toroidal, (d) Ferrite-core, (e) Unshielded SMD, and (f) Shielded SMD.

Together with resistors and capacitors, inductors belong to the same passive linear circuit elements found in electrical circuits. Inductors and capacitors are two types of energy storage elements—the former deals with current while the later with voltage. On its own, inductors are used extensively in analog circuits and signal processing applications; when teamed up with capacitors, they form tuned circuits that either emphasize or filter out specific frequencies or signals.

Inductors come in various shapes and sizes as well. Some are quite similar to resistors in that they have the same color bands, and you can easily mistake them for resistors unless you measure them.²⁴ The color code works the same way with inductors, except the values are in micro-henries (μH). For SMD inductors, the values are usually marked on top as a 3-digit numerals, the first 2 digits are numbers and the third a multiplier, so $101 = 10 \times 10^1 = 100\mu\text{H}$, and $1R5 = 1.5\mu\text{H}$ (where R acts as a decimal point).

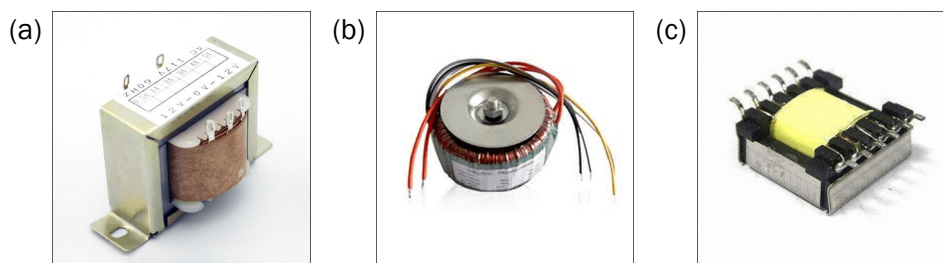
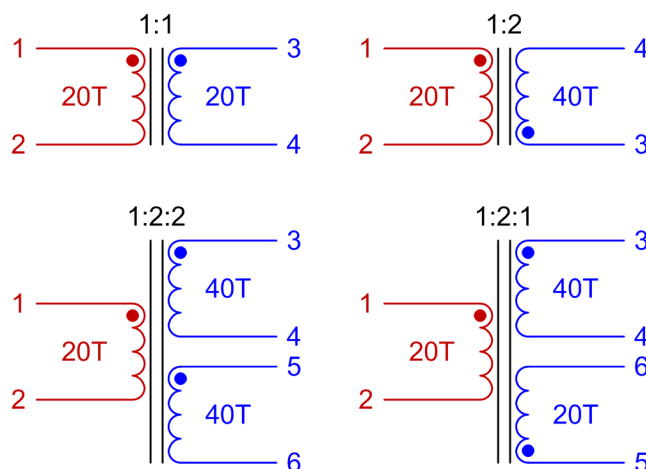


Figure 2.8 Types of transformers: (a) Center-tapped, (b) Toroidal, and (c) Multiple windings.

Transformers²⁵ are variations, or should I say, combinations of inductor coils that produce two or more windings around a common core. These are usually used in power supplies to provide step-up or step-down of voltages, in EL lamp driver circuits as flyback transformers, or in the case of communications circuits, as matching transformers. In most cases, the model or part numbers are printed on the casings so it's not a problem referencing the electrical specifications off their datasheets. Sometimes, the transformers may be custom-made so information is unavailable. This isn't an issue if you only need to draw the electrical representation; you just need to make sure the topology is correct.



Examples of transformer winding configurations

²⁴ Generally inductors exhibit near short to a few ohms, though there are custom wound types that may give much higher values.

²⁵ I once asked my nephew what is a transformer and he replied, "Robots from the planet Cybertron, you know—Autobots and Decepticons." Yeah right, more than meets the eye!

FUSES

A fuse is a kind of resistor with low resistance and built to tolerate a certain amount of current, which if exceeded, will cause it to blow up and become an open. In other words, it's a protection device.

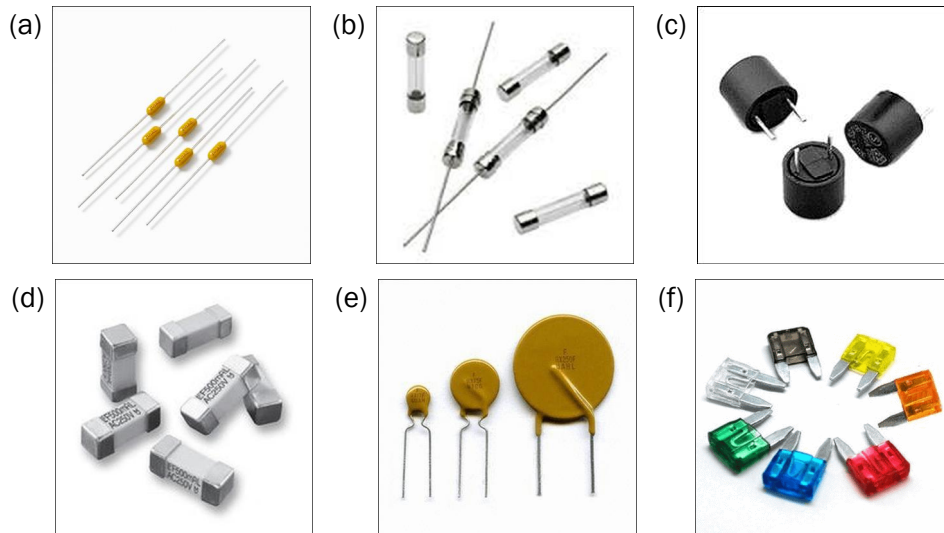


Figure 2.9 Types of fuses: (a) Axial-lead, (b) Glass cartridge, (c) Radial, (d) SMD, (e) Resettable, and (f) Blade.

The most commonly seen fuses are the glass cartridge type with metallic caps at both ends that snaps firmly onto a fuseholder.²⁶ This allows for easy replacement when a fuse blows due to an overcurrent condition. Fuses come in different shapes and sizes as well, axial and radial, through-hole and SMD, you name it and you'll most certainly find it.²⁷ What's more, there now exists a kind of resettable fuse that opens upon overcurrent condition and recovers when the current returns to normal.

Fuses have two ratings: voltage and current. Most engineers (and electricians) think that the current rating is more important than the voltage rating. Not necessary. While it is important to select the right current rating to ensure a fuse adequately protects a circuit without blowing off under normal operating conditions, having the right voltage rating does matter as well. Not so much a matter of AC or DC, but whether it leads to arcing when a fuse blows.²⁸ Therefore, a higher voltage rated fuse can be used to replace a lower voltage rated fuse, as long as the current rating is the same, but not the other way around...

And if I can't convince you, hopefully I've managed to confuse you! (con-fuse, get it?)

²⁶ These fuses are mostly located at the primary side of power supplies at the AC live input line, providing a front-line protection against catastrophic failures.

²⁷ Major manufacturers include Littlefuse, Picofuse, Bussman, TE Connectivity, etc.

²⁸ Fuses rated above 600V are ceramic insulated to withstand heat and stress arising from the higher voltages.

RELAYS AND SWITCHES²⁹

Make or break, the relay is here to stay. Some examples are shown in the figures below:

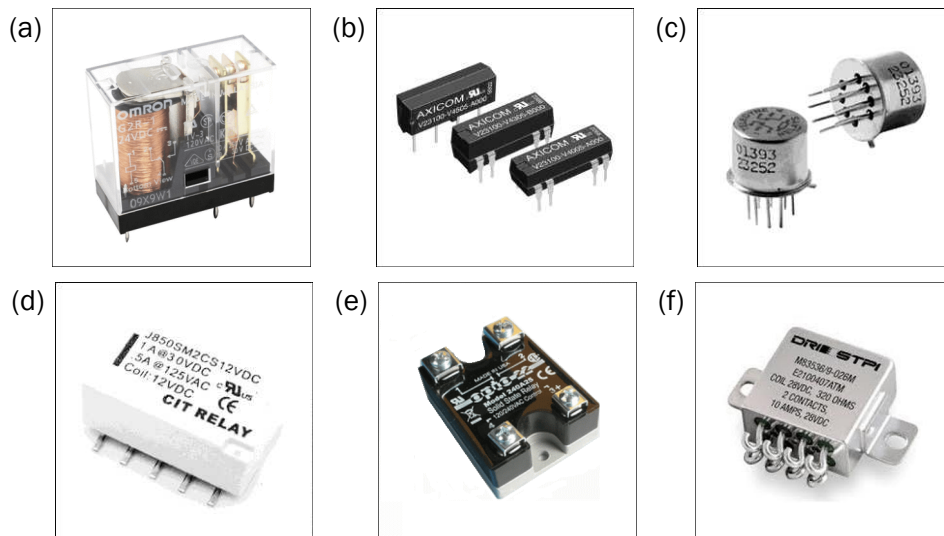


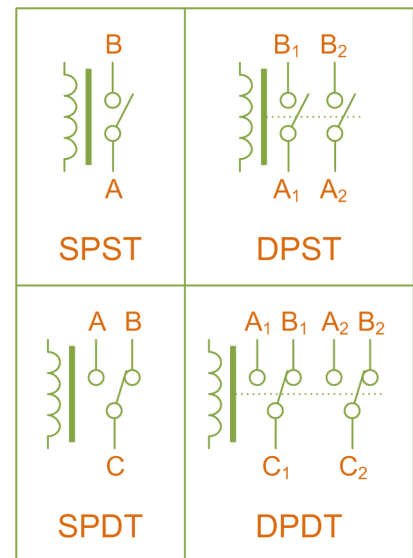
Figure 2.10 Types of relays: (a) Electromechanical, (b) Reed, (c) TO-5, (d) SMD, (e) Solid-state, and (f) Hermetically sealed.

Most relays are electromechanical in structure in which contacts make-and-break by means of a relay coil being energized or de-energized by a control voltage. For this reason, they are also called electrically operated switches.

Standard relay configurations are (refer to right figure):

- SPST Single-Pole Single-Throw
- SPDT Single-Pole Double-Throw
- DPST Double-Pole Single-Throw
- DPDT Double-Pole Double-Throw

You may have heard people mentioning form A, B, C or D relays and wonder what all these terms meant. Well, it has to do with the way the contacts make or break. Form A relays have normally open (NO) contacts which make when activated and break when inactive. Form B relays are the opposite, that is, they are normally closed (NC) when inactive and break when activated. Form C are change-over (CO) or double-throw (DT) relays with a common terminal that connects to a normally-closed contact when inactive, and switches to the normally-open contact when activated. This is termed break-before-make. Form D relays are make-before-break.



²⁹ Technically speaking, relays are switches, so terminologies that apply to switches are applicable to relays.

Solid-state relays (SSRs), on the other hand, are made up of semiconductor materials, and therefore possess very fast switching speeds (nanoseconds) without the wear and tear of electromechanical relays. However, they have limited switching configurations (mainly SPST switching), and the inherent higher ON resistance produces heat as well as electrical noise.

Switches are similar to relays in that they operate by make-and-break of contacts, except that they are manually set to a fixed state or position(s).

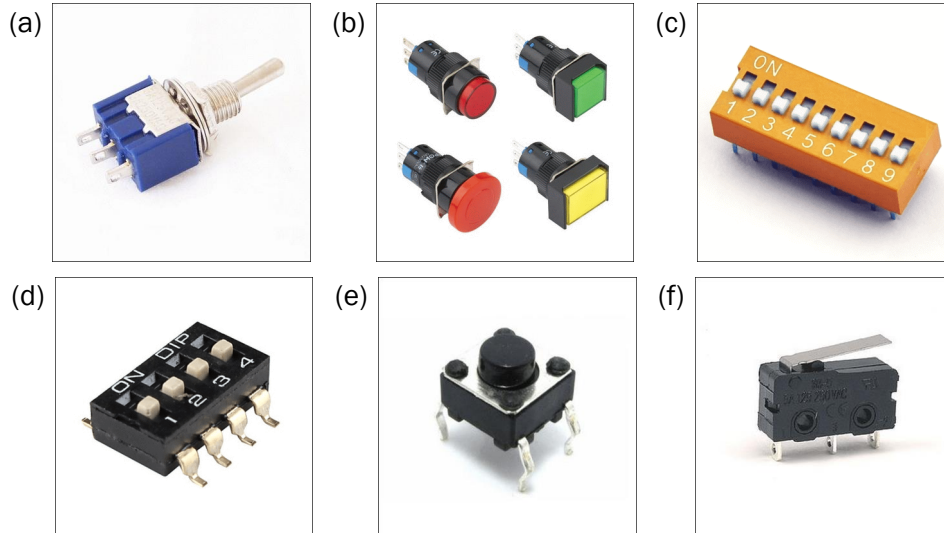


Figure 2.11 Types of switches: (a) Toggle, (b) Pushbutton, (c) DIP, (d) SMD DIP, (e) Miniature Pushbutton, and (f) Micro-switch.

While switches are generally used to turn electric circuits on and off, they have diversified to different types that serve other purposes: toggle switch for resetting, pushbutton switch for keypads, DIP switch for configuration settings, micro-switch for limit detection, etc.



Of special mention is the two-terminal reed switch³⁰ that is hermetically sealed in vacuum or inert gas within a glass capsule. This type of switch is small and light-weight, can operate at high switching voltage ($>1000\text{V}$) yet able to switch small currents (femto-Amp), and works reliably (up to 10^8 cycles) under harsh environment (extreme temperatures and vibrations). Reed switches are used in telecom, medical electronics, test equipment, automotive, and power industries, etc. It's even found in the power supply test station in my work lab!

³⁰ The reed switch is more of a hybrid between a switch and a relay, since it depends on an external magnetic source to activate the contacts within the glass capsule. It is, however, different from a reed relay which is electro-mechanically self-contained.

CRYSTALS AND OSCILLATORS

Before going into active components (semiconductors and ICs), let's take a look at one type of device that is widely used in many digital and processor-based PCB designs—the crystal oscillator.

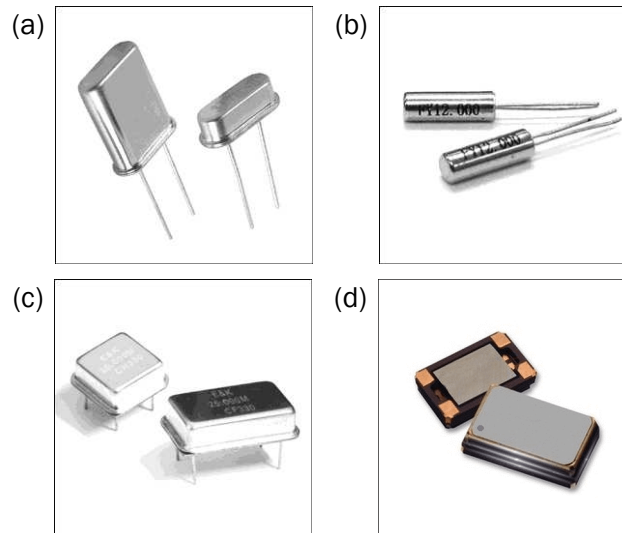


Figure 2.12 Types of crystals: (a) Discrete, (b) Quartz, (c) DIP oscillator, and (d) SMD oscillator.

Crystal oscillators, whether excited by the host ICs which they are connected to, or powered by external sources, exhibit high quality factors (Q) above 10,000 and are suitable for use as resonators and in high stability tuned circuits.

In its simplest form, a crystal oscillator has a fixed frequency vibration with accuracy up to $\pm 30\text{ppm}$, but greater accuracy and stability can be further achieved in mission critical system using special variants like the voltage-controlled crystal oscillator (VCXO), temperature compensated crystal oscillator (TCXO), and the oven-controlled crystal oscillator (OCXO).

Some components have built-in crystals, such as the real-time clock (RTC)³¹ made by Dallas Semiconductors³² to provide accurate 24-hour time in applications like the personal computer (PC) and embedded systems.



³¹ The oscillator's frequency is usually 32,768 kHz. Some RTCs use the power line frequency as reference instead. These RTCs have embedded batteries to sustain in-built CMOS memories to keep time running even when the system is powered off, and can usually last for 5 years or longer.

³² Other manufacturers are Benchmarq, STMicroelectronics (not related to my company!), and Maxim, etc.

DIODES AND ZENERS

Nowadays when we talk about diodes, we think in terms of semiconductor (germanium or silicon) type and make. The younger generation of engineers probably may not have even heard of the thermionic diode, a vacuum valve device that looks very much like an incandescent light bulb.³³ But I'm not about to get nostalgia here.

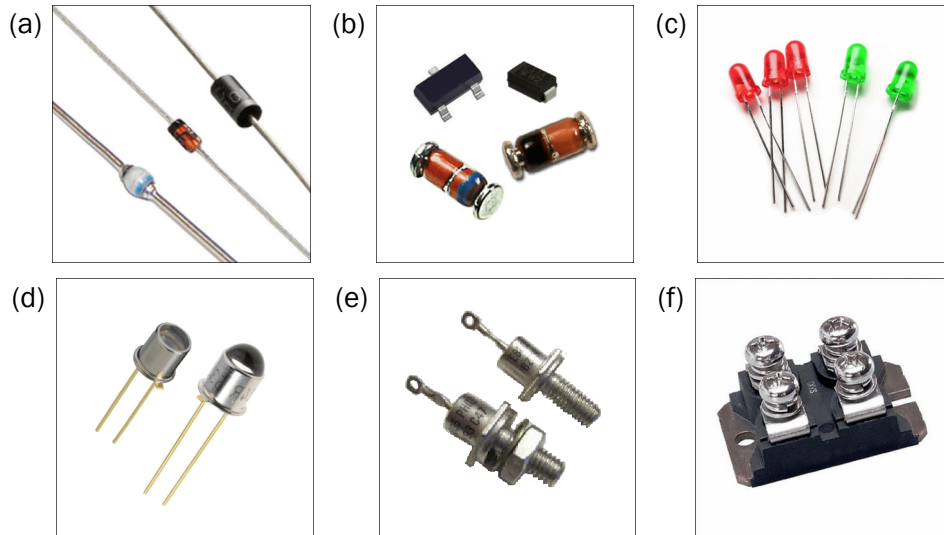


Figure 2.13 Types of diodes: (a) Axial lead, (b) SMD, (c) LED, (d) Photodiode, (e) Screw-mount power, and (f) IGBT.

A diode, by definition, is a two-terminal non-linear device that works on the p-n junction principle that exhibits forward and reverse biasing characteristics. There are many types of diodes, from the common rectifier to the varactor, the light-emitting to the photo-laser,³⁴ and the step recovery to the zener, etc.

So what is the difference between a diode and a zener?

While a diode operates on the forward voltage region and may suffer catastrophic failure if subjected to voltages that exceed its reverse breakdown limit, the zener can safely operate on the reverse voltage region. In other words, while a diode conducts current in only one direction, a zener, by virtue of its heavy narrow junction doping, can conduct current in both directions with the ability to maintain a stable voltage in the reverse bias condition. This makes the zener ideal for applications such as voltage references in comparator circuits and voltage stabilizers in low-current circuits.

³³ In my secondary school laboratory, there was this old model oscilloscope that run on vacuum tubes. And in my polytechnic curriculum, I studied the diode and triode valves in basic electronics as well as television theory. To top it off, my family once owned a TV that had these legacy components inside too! Of course, those into hi-fidelity sound systems still swear by these devices, claiming they produce the best sound quality ever.

³⁴ Light-emitting diodes (visible and invisible types) and optocouplers all come under optoelectronics, together with fiber-optics.

How then do you differentiate a zener from a diode? Apart from the part numbers, it would not be obvious from the physical appearance, though zeners mostly come in opaque coatings compare to the more transparent glassy form of the diodes for axial lead packages, while some zeners have a blue cathode stripe instead of the usual black for a diode.



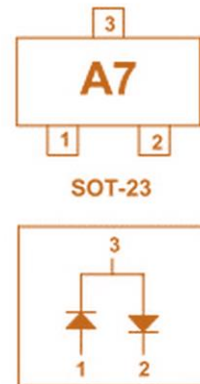
There is no hard and fast rule, but the following suggestions may help:

- Check the obvious—zeners have reference designator prefixes such as Z or VR
- Some zeners have their zener voltage values printed on their bodies
- Google the part number
- Look out for protrusions near where the body and terminals merged
- Infer from the way it is connected in the circuit³⁵

Military grade diodes are generally prefixed with JAN, JANTX, JANTXV, or simply JX before their usual 1Nxxx commercial part numbers. The JAN prefix indicates that the device is specially manufactured for joint army and navy applications, and can be specifically certified to aerospace quality levels. A useful point to take note when working on military PCBs.



Due to their small sizes, SMD diodes are usually marked with 2 or 3 alphanumeric codes. For example, on the right are two SOT-23 diodes marked A7W and 87. The first is the SMD code and the second is the year of manufacture. There are online³⁶ look up references for these discrete SMD components that you can easily find. (See [Appendix A-4](#) for a sample page.) In our case, the SMD code A7W references to BAV99, a dual diode in SOT-23 package made by Philips Semiconductors.



As you'd probably realized by now, unlike the axial-lead diodes, SMD packages can contain more than one diode per package type, such as the BAV99 example just mentioned. Things can get really tricky, however, when you work on PCBs with many of these SMD type components present, so it's good to download a copy of the SMD Codebook³⁷ and keep it handy for quick reference.

³⁵ Most of the time there's no need for such measures, unless the component is SMD type without any marking or designator to identify it. In such instance, I'll usually attempt a partial reverse engineering around that component to see how it's connected to determine if it's a diode or zener.

³⁶ One of the most comprehensive and regularly updated site: <http://www.marsport.org.uk/smd/mainframe.htm>

³⁷ There are a number of free pdf electronic version of SMD Codebooks downloadable from the internet which cover a broad enough range of diodes and transistors. But if you're willing to fork out about 22 euros, you can get a really good one from www.turuta.md which covers over 300,000 active SMD semiconductor components in their 2014 edition.

TRANSISTORS AND MOSFETS

The transistor—from its humble beginning as a discrete entity in 1947 at AT&T's Bell Laboratory, to its present highly integrated forms numbering in the millions—is the basic building block of all modern electronic devices.

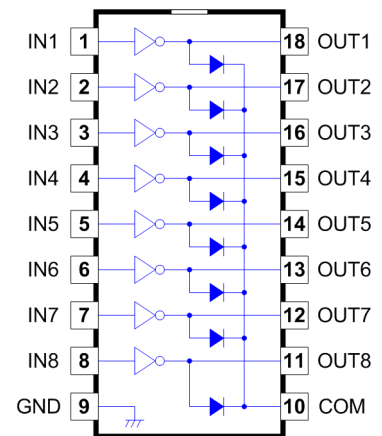


Figure 2.14 Types of transistors: (a) BJT/JFET, (b) Power, (c) MOSFET, (d) SMD, (e) Hermetically-sealed SMD, and (f) Integrated.

Transistors can be broadly classified into two types: bipolar and unipolar (also known as field-effect transistor, or FET). The bipolar transistor³⁸ has three terminals labeled base, collector and emitter, and utilizes a small base current to control or switch a much larger current between the collector and emitter. As such, the BJT can function as an amplifier in the linear range, or as a switch under saturated condition. The FET also has three terminals labeled gate, drain and source, but uses a gate voltage to control the current between the source and drain. The FET can be further divided into junction FET (JFET) and insulated or metal-oxide semiconductor FET (MOSFET).

Like the diodes, military grade transistors are similarly prefixed with JAN, JANTX, JANTXV, or simply JX before their usual 2Nxxx commercial part numbers. Likewise, SMD transistors are marked with 2 or 3 alpha-numeric codes. Figure 2.14(d) above is a Fairchild MMBT2222A, the equivalent of a 2N2222A TO-18 npn transistor. There are also transistor array ICs used mainly for driving/sinking high current loads, such as the ULN2803A shown on the right.

[Appendix C-11](#) outlines the various transistor packages and pin-outs.



³⁸ Also known as bipolar junction transistor, or BJT in short. It's termed bipolar because a BJT is made up of two p-n junctions formed side by side to produce a sandwich n-p-n or p-n-p layer combination.

INTEGRATED CIRCUITS

Integrated circuits (IC) are by far the most diversified and interesting of all electronic components, from small- to medium-scale (SSI/MSI) integration like the TTL/CMOS chips, to the large and very large-scale (LSI/VLSI) peripherals and microprocessors, and finally to the ultra large-scale (ULSI) containing over one million transistors, and the wafer-scale integration (WSI) that uses the entire silicon wafer to fabricate a super-chip.

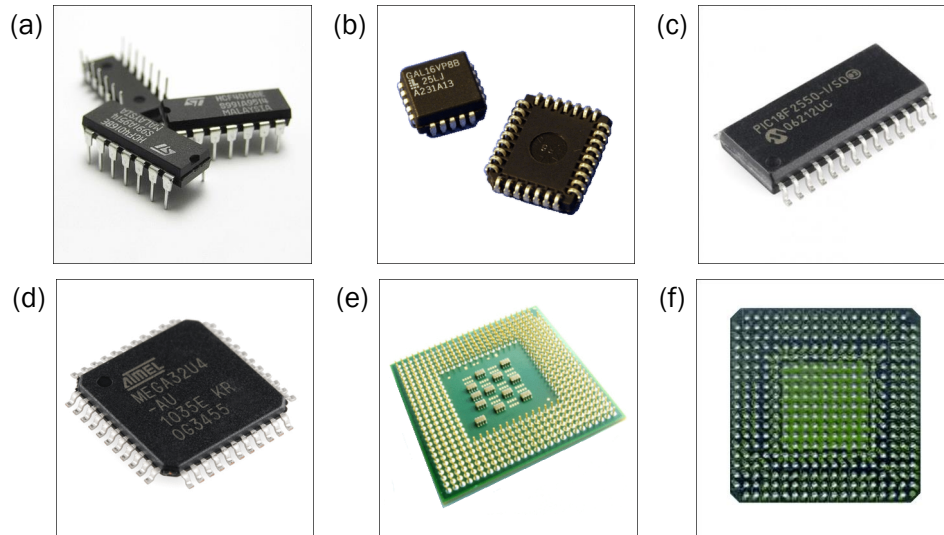


Figure 2.15 Types of ICs: (a) DIP, (b) PLCC, (c) SOIC, (d) QFP, (e) PGA, and (f) BGA.

And if you think you've seen enough possible varieties of component packages in the earlier sections, think again! As of April 2014, the existing list of integrated circuit packages can be grouped under eight major categories³⁹:

- | | |
|-------------------|--|
| ▪ Through-hole | SIP, DIP, CERDIP, QIP, SDIP, ZIP, MDIP, PDIP |
| ▪ Surface mount | CGA, CCGA, CQGP, LLP, LGA, MCM, Micro SMDXT |
| ▪ Chip carrier | BCC, CLCC, LCC, LCCC, DLCC, PLCC |
| ▪ Pin grid array | OPGA, FCPGA, PAC, PPGA, CPGA |
| ▪ Flat pack | CFP, DFN, QFN, PQFN, TQFN, QFP, BQFP, CQFP, LQFP, PQFP, TQFP |
| ▪ Small outline | SOIC, SOP, CSOP, MSOP, PSOP, QSOP, SSOP, TSOP, TSSOP |
| ▪ Chip-scale | CSP, TCSP, TDSP, COB, COF, COG, Micro SMD |
| ▪ Ball grid array | CBGA, FBGA, LBGA, PBGA, SBGA, TBGA, μ BGA |

The above list is by no means exhaustive, but it gives you an idea of the extent the world of integrated circuits encompassed. [Appendix C-1](#) contains sample isometric illustrations of popular IC packages which might be useful in identifying these myriads of ICs you may come across on a PCB. Of course, it does help to know that ICs are mostly designated as U or (you guess it) IC on the PCB's solder mask.

³⁹ Source: Wikipedia, under List of integrated circuit packaging types.

Apart from their packages, ICs can also be categorized under the type of functions they perform, being classified broadly under digital, analog, and mixed signal. We will briefly mention a number of them in the following sub-sections:

Digital Logic

These are the basic building blocks of digital logic circuits—digital because they operate on binary states of 1's and 0's, and logic because they follow Boolean rules or truth-tables. Common elements of logic include gates (AND, NAND, OR, NOR, XOR, XNOR, and NOT), buffers and inverters, flip-flops (R-S, D, and J-K types), adders and comparators, shift registers and latches, encoders and decoders, multiplexers and de-multiplexers, etc. Examples are the 74-series TTL and 4000-series CMOS ICs.⁴⁰ These logic elements can be used to construct two types of logic circuits, combinational and sequential.

Memories

Data storage components formed one of the major group of integrated circuits. Generally they can be divided into read-only (ROM) and read-writable or random access (RAM) memories. Basically, a ROM will retain its data once it is written, and may no longer be altered if it's a one-time programmable (OTP) type, or can be re-programmed if it is of the erasable-programmable type such as the EPROM, EEPROM, or FLASH. RAM, on the other hand, is much like a scratch pad and retains its data so long as power is present, and loses its data once power is turned off.

RAMs come in two flavors—static (SRAM) and dynamic (DRAM). A static RAM's memory cell is usually a flip-flop and can be set or reset to represent either logic 1 or 0; a dynamic RAM's memory cell is simply made up of a transistor and a capacitor to hold one bit of data, but requires periodic refreshing to retain the data. Therefore, for a given area of silicon real estate, a DRAM can hold more data bits than an SRAM, yet consume considerably less power, the trade-off being the need for refresh memory control circuits and slower access speed.

The advent of electrically erasable memories such as the EEPROM (or E²PROM) and FLASH has changed the memory landscape and introduced what is termed solid-state memories as alternative replacements for the conventional bulky hard disk drives.⁴¹ FLASH are generally more versatile than the EEPROM, and faster because of their larger block sizes when erasing and writing large amount of data. The downside is these memory devices have limited number of write-erase cycles (typically 100,000 before data integrity becomes an issue). However, technology is improving and gradually diminishing this limitation.

⁴⁰ TTL stands for transistor-transistor logic, and CMOS stands for complementary metal oxide semiconductor (or MOS). [Appendix D-1](#) and [D-2](#) give the pinout diagrams for some of the common 54/74-series TTL ICs, and [Appendix D-3](#) and [D-4](#) the pinouts for the 4000-series CMOS ICs.

⁴¹ While solid-state drives (SSD) are 3 to 4 times more expensive than a hard disk drive (HDD), they perform much faster. I recently upgraded my laptop to a Samsung 256GB SSD and Microsoft Windows starts up and shuts down in less than a third of the time it took with a HDD. Even applications run noticeably much faster too!

Programmable Logic

A programmable logic device (PLD) is very much like a digital logic black box that enables a hardware designer to define inputs and outputs using Boolean (or logic) equations, or finite state-machine tables. PLDs have come a long way—from the humble sum-of-product terms PALs, to the electrically erasable-programmable generic GALs, to the complex sea-of-gates CPLDs, and finally to the highly sophisticated FPGAs.

PAL—Programmable array logic was first introduced by Monolithic Memories Inc. or MMI in 1978 and quickly gained acceptance among designers, replacing the combinational-only limitation of using ROMs as straight look-up tables (LUTs), and improving on its cumbersome predecessor, the PLA. Besides buffered programmable outputs such as the 16L and 20L families, PALs also come with registered data outputs like the 16R and 20R families as well.

GAL—Generic array logic, invented by Lattice Semiconductor in 1985, soon became the popular choice for design prototyping due to its re-programmability and the ability to emulate a wide range of the 16- and 20-series logic/registered OTP PALs. Hot favorites are the 16V8 and the 20V8 GAL chips.

The PAL—Teaching Old PROMs New Tricks



CPLD—Complex programmable logic devices can be viewed as the big brother of PALs and GALs since it is the equivalent of several PALs linked by programmable interconnections inside one chip. The basic building block of the CPLD is a macrocell (which is more complex than the prefabricated array of simple logic gates found in PALs) made up of high level logic elements like flip-flops, registers, and even ALUs. A CPLD's gate density, such as that of the Altera MAX's 3000, 5000 and 7000 families, can easily range from several thousand to a few hundred thousand.

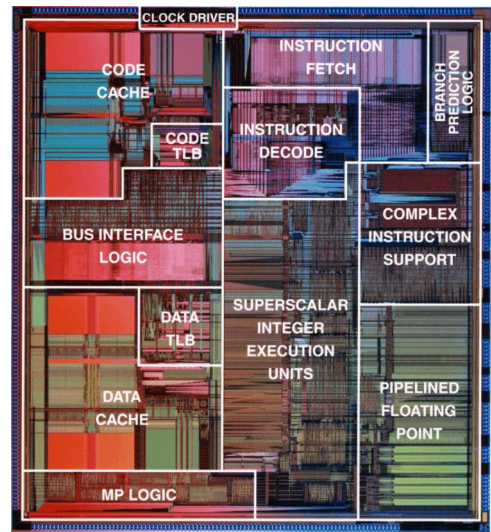
FPGA—Field-programmable gate array is made up of an array of logic blocks⁴², I/O pads, and routing channels. Within each logic block are several logical cells, each comprising an LUT, a full-adder, a D-type flip-flop, and multiplexers. Unlike the non-volatile programmable nature of CPLDs, FPGAs are generally SRAM-based which require them to be programmed by external storage devices such as configuration PROMs upon power up, and lose their configuration data when power is removed. There are a suite of tools that allow a designer to define the behavior of the FPGA, though most of these software utilize either hardware description language (HDL) or schematic entry as the basis for design entry. Due to their different architectures, CPLDs are better suited for wide combinational logic applications, whereas FPGAs are good for implementing large state machines such as processors.

⁴² Depending on the vendors, these logic blocks are called Configurable Logic Block (CLB, Xilinx) or Logic Array Block (LAB, Altera). FPGA vendors give fanciful names to their products too—Altera has its Stratix, Arria and Cyclone series, while Xilinx has its Spartan, Artix, Kintex, Virtex, and more recently the Zynq-7000, which is closer to an SoC (system on a chip) than an FPGA.

Processors (CPU)

A processor or CPU is the brain of a system or board. Today, CPUs are no longer limited to just the basic microprocessors requiring additional support from peripheral components to extend their functions and usefulness, but include variants like microcontrollers (MCUs), digital signal processors (DSPs), bit-slice processors, graphics processing units (GPUs), multi-core processors, and system-on-a-chip (SoC) processors. Despite their numbers and varieties, processors are generally based on either complex instruction set computing (CISC) or reduced instruction set computing (RISC) architecture.

On the right is a magnified digital image⁴³ of Intel's second generation Pentium die layout with its 3.3 million transistors, showcasing the major building blocks that make up a modern microprocessor chip.



Peripherals and Chipsets

CPUs don't exist alone, though in small embedded designs and applications, a microcontroller might just make do with minimum supporting logic circuits. In most instances, however, additional components known as peripheral ICs or chipsets are necessary to offset the workload of the CPU so it can focus on its essential tasks, while at the same time providing the CPU a bridge or interface to the real world so it can interact with humans.



In the era of the 8080 and 8085 microprocessors, Intel provided the bulk of peripheral ICs such as the 8254, 8255, 8259 and 8279 etc. When other chip manufacturers began making their own CPUs, they too introduced compatible peripheral ICs to complement their flagship products.⁴⁴ And as Moore's law predicted, the complexity of integrated circuits doubles at a rate of two years, microprocessors and their peripheral chipsets have also advanced beyond their humble origins.

To support advanced microprocessors like the Pentium and higher, Intel has come up with new classes of chipsets termed the Northbridge (high-speed interface) and the Southbridge (lower speed peripheral bus interface). Other vendors like SMSC also produced simpler chipsets such as the Super I/O Controller (see inset figure above).

⁴³ Source: Computer History Museum

⁴⁴ One of Intel's competitors in the 8-bit microprocessors business was Zilog, who came up with the Z80 CPU and a slew of peripherals such as the CTC, SIO, DMA, PIO, and DART. My third year in tertiary, we studied the Motorola 6502 but I was more interested in the Z80 so I ended up building a project using it and as a result studied two microprocessors instead of one. Looking back now, am I crazy or what?

Analog (Linear)

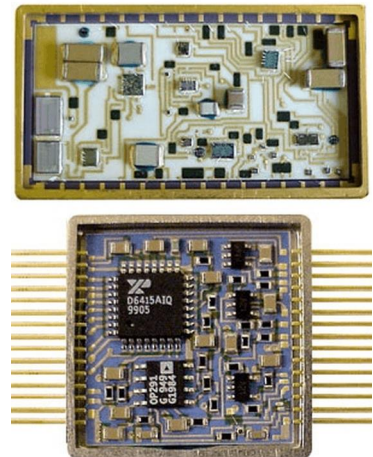
Unlike digital ICs which operate on discrete signals such as the binary 1s and 0s, analog or linear devices operate on time-varying continuous signals that are unipolar (0 to supply voltage) or bipolar (negative to positive supply voltages). Due to the inherent nature of their functions and performance specifications, analog IC designs do not progress in phenomenon rates like their digital counterparts, are less densely packaged yet demanding on the die layout requirements.

Examples of analog ICs are operational amplifiers, analog switches, analog multiplexers/de-multiplexers, PWMs, regulators, active filters, comparators, analogue-to-digital (ADC) and digital-to-analogue (DAC) converters, etc. They are mainly used in applications such as signal processing, data acquisition, power and servo controls, instrumentation and communication systems, etc.

Hybrids

Hybrid microelectronics are a class of integrated circuits that combine many electronics components in a miniature PCB-like substrate that fits snugly within a customized integrated package. Hybrid devices come in different shapes and packages; some are even epoxy-encapsulated to protect them from the environment they operate in, or to provide structural support due to the way the components are mounted.

On the right are two examples: (Top) a servo loop containing eight ICs, two MOSFETs, several SMD resistors and capacitors housed in a 32-pin ceramic package; (Bottom) a surface-mount flat pack containing five ICs, 21 resistors, 16 capacitors, hermetically sealed for reliability.



(Courtesy of Sanchez Microelectronics, Inc.)

Military Part Numbers

It is not uncommon to find ICs on military PCBs with part numbers beginning with M38510 instead of the normal industrial or commercial part references. For example, the 7400 quad 2-input NAND gates is marked as M38510/00104 on the package—not very helpful in identifying it as a 7400. Fortunately, there is an online service provided by the US Defense Logistic Agency (DLA), allowing you to cross reference these cryptic JAN/SMD numbers to their actual vendor part numbers.⁴⁵ I have also included the cross references for some common JAN/SMD numbers in [Appendix A-7](#) thru [A-9](#). When filling up the BOM, you can choose to retain the JAN/SMD numbers, replace them with their vendor part numbers for easier reference, or include both.

⁴⁵ Website: <http://www.landandmaritime.dla.mil/Programs/Smcr/> (Not sure if it will stay active indefinitely.)

MISCELLANEOUS

The world of electronics is not only vast, it is expanding everyday as new devices and gadgets are being invented and produced. And while new products are getting more compact and complex functionally, the number of components used are getting fewer, more integrated and modular.

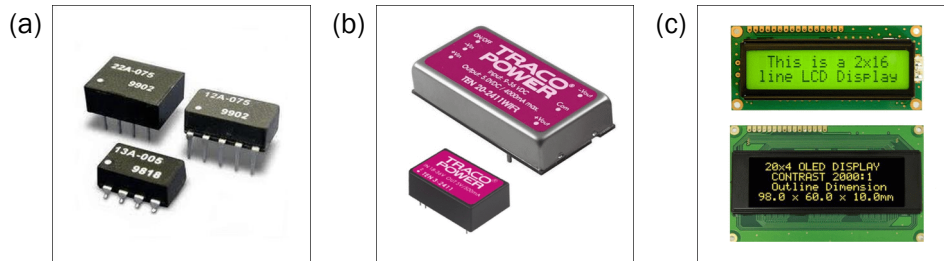


Figure 2.16 (a) Delay lines, (b) DC-DC converters, and (c) LCD and OLED modules.

The above are some examples and are by no means exhaustive; these are grouped under miscellaneous since they are modular and do not fit nicely into the earlier categories of components. Some others that I can think of are filters, power dividers, modulators and demodulators, sensors and transducers, etc. I'll briefly discuss two devices below:

Delay Lines

A delay line device is used to slow down a signal by a time interval from its input to its output. Generally, these devices are classified under passive and active, and the delay interval can be a fixed value or variable (i.e. programmable). Passive delay lines or analog delay lines (ADL) are made with analog components such as capacitors and inductors. Though ADLs can delay both analog and digital signals, they tend to attenuate and distort the signals they delay. Active delay lines or digital delay lines (DDL) are made using digital components (TTL, CMOS, or ECL logic) and can only be used to delay digital signals. Unlike ADLs, DDLs do not attenuate or distort the signals.

DC-DC Converters

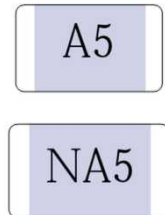
Most power supply modules in the market convert voltages from AC to DC and vary in sizes depending on the power wattages. These are usually standalone units that can be connected to the system or board they are powering via wired connectors. A DC-DC converter, on the other hand, converts a DC voltage to one or more DC voltages and polarities, and can be small⁴⁶ enough to be mounted on a PCB to provide specific voltages to selected portion of the board circuitries.

⁴⁶ Higher power wattage models such as those manufactured by Vicor or Interpoint can either be chassis-mounted or PCB-mounted (if space permits). I have encountered both types in my work.

COMPONENTS WITHOUT MARKINGS

As the size of components get smaller, it becomes impossible to label them intelligibly, or even to label them at all.

Some manufacturers do provide laser markings on their chip capacitors to prevent surface degradation or induced micro-cracks; these codes are based on the EIA RS 198 standards which use 2- or 3-digit alphanumeric codes to represent capacitance values. With these codes, you can then look up their product datasheet reference to determine the values (see [Appendix A-6](#)). For example, A5 = 100,000pF or 100nF, where N denotes the vendor NovaCap.



But what do you do with SMD components without markings?

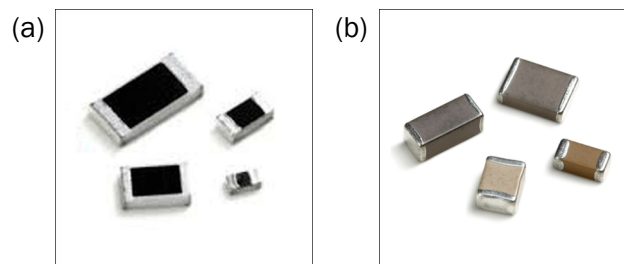


Figure 2.17 SMD resistors and capacitors without markings.

While you're more likely to encounter chip capacitors with no markings than chip resistors,⁴⁷ resolving the latter is a much simpler and straightforward affair. A normal digital multimeter can usually measure resistors in-circuit quite accurately without interference from adjacent components,⁴⁸ unless there are parallel resistors or low impedance path across the resistor in question.

Unfortunately there's no way you can measure a capacitor in-circuit due to PCB track capacitances as well as induced lead and bond-wire capacitances of components around it. For the most part, you can get some hint from the usage of the SMD capacitors:

- De-coupling capacitors across the power and ground pins of digital ICs tend to be 100nF
- Crystal load capacitors for processor clock come in pairs and are either 15pF, 22pF or 33pF (the CPU's datasheet application notes will usually state the preferred value)
- Some analog or mixed ICs (ADCs and DACs) datasheets will also provide the values of capacitors to be used for voltage or current compensation, power boosting purposes, etc.

⁴⁷ It would seem SMD resistors are made of materials suitable for printing, while most SMD capacitors are ceramic material which requires an extra coating to reliably print on. Higher cost of production and impact on quality of the product may be the reasons for not marking them at all.

⁴⁸ Analog meters are not suitable for in-circuit resistance measurement due to the high voltage output (between 3-12V) which will turn on nearby semiconductor devices and produce wrong readings. Digital multimeters usually have less than 0.6V output in comparison. (Subject to resistance values <200k-500k.)

We can see that datasheets do play an important role here, though not always.⁴⁹ In the course of my 15 years of reverse engineering, I've had not bothered to remove those SMD capacitors and have them measured with an LCR meter. It's too much hassle and do not really serve useful purpose, in my humble opinion. However, if you're a perfectionist and can't stand having your schematic drawings populated with valueless components, then have it your way (grin!).

MISSING REFERENCE DESIGNATORS

Sooner or later, you are going to come across PCBs with components that do not have their reference designators printed on the silkscreen layer, either due to congestion of space or by design. What then?

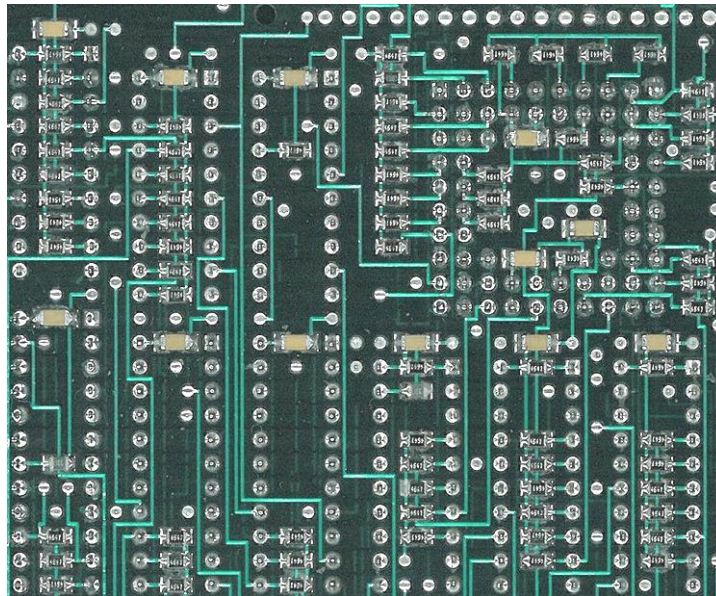


Figure 2.18 Components without reference designators.

In such instances, may I suggest that you do the following:

Step 1:

Take note of component designators that are available on the component and solder sides' silkscreen layers of the PCB. Resistors and capacitors are usually marked with R and C followed by running numbers, while diodes may be D or CR, and integrated circuits prefixes with U or IC. Table 2.2 lists the prefixes commonly associated with their related components. Note down the largest number for each of the group designators (e.g. R123, C99, U68, etc.).

⁴⁹ In the last section of this chapter, we will see why component datasheets are one of the contributing factors to the success of doing PCB reverse engineering, besides possibly helping you to determine the values of components without markings.

Step 2:

Determine the arrangement or layout of the components, paying attention to the way the reference designators run on the PCB, horizontally or vertically, from left to right or vice versa. Then follow the flow by adopting either a row or column numbering pattern accordingly.

Step 3:

Make a photocopy scan⁵⁰ of the PCB, grayscale and inverse it, and then segment it into grids as shown below. Start numbering the unmarked resistors based on the grid reference: R1A, R1B, R1C, etc.

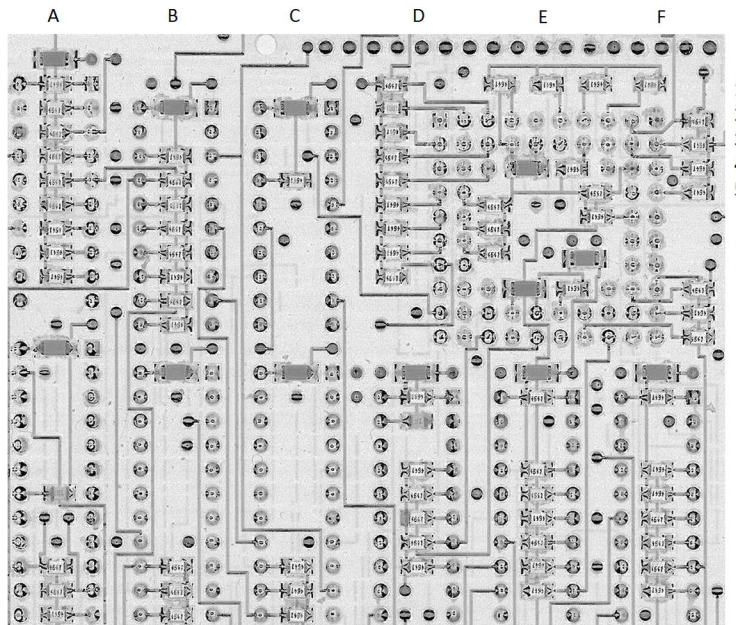


Figure 2.19 Segmented PCB with grid references.

The above example uses column numbering pattern due to the layout of the components which makes it suitable for this scheme. Sometimes the components' layout maybe haphazard or random, in which case you can still produce a grayscale-inverse artwork of the PCB, print it out and manually write on the hardcopy. This is the fastest way to assign designators, but is quite messy and error prone.

A better way is to wait till you complete drawing the Visio layout diagram of the PCB, as outlined in the next chapter, before assigning the designators. This will save you a lot of trouble in case you miscount or miss-count the components and need to white-out or erase the errors and re-number the designators on paper.

⁵⁰ Nowadays modern photocopiers are multi-functional type with scan function that allows you to make a photo copy of the PCB literally. Alternatively, you can use a camera to take a photo and copy it into your PC for further image processing.

A picture speaks a thousand words, so below is the Visio artwork of the above PCB portion, just to give you an idea what I meant:

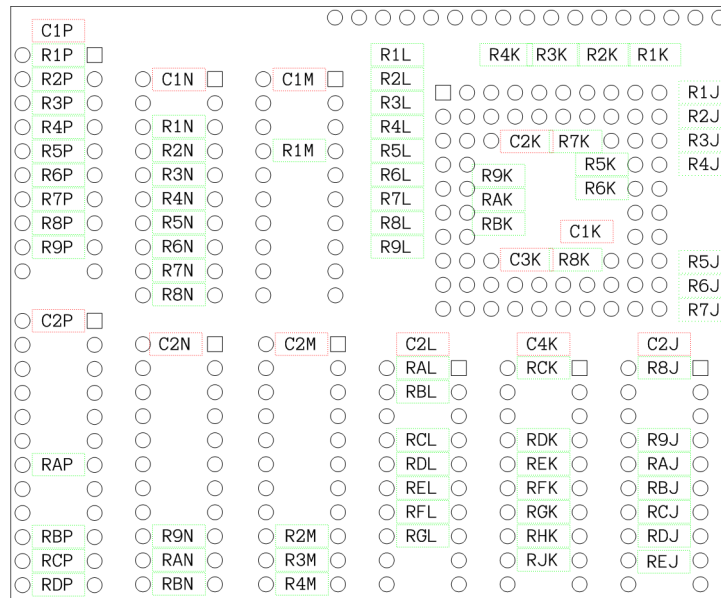


Figure 2.20 Components assigned reference designators.

Looks much neater, doesn't it?

FILLING UP THE BOM

Once you get the ground work of filling up the bill of materials (BOM) done, you'll start to have a better picture of the PCB you're going to work on. It's far from over, though. Organizing the BOM will take some thought and effort to ensure it is easy to refer to later on.

The sample BOM which I provided in Table 2.1 places the part number as the reference index with respect to the component designator. This is space saving but it has a problem—if you need to know what U16 is, you'll have to go all over the IC group to locate U16 before you can find out that it is a LM117H. Another way of doing BOM is to place the component designator in alphanumeric order as the reference index with respect to the part number. This means that if there are 100 ICs, you'll end up with 100 lines since the list will start from U1 all the way to U100, with multiple repetitions of the same part number for certain oft-used components albeit in different lines. This way, if you want to know what is U43, you'll get it in quick time going down the designator index, that it's a 54FCT373.

Both ways of representing the BOM have their pros and cons, so I'll leave it to you to decide which suits you best.⁵¹

⁵¹ You can even maintain two separate BOMs if you like! I usually do that if the PCB has lots of components on it—it's hard work I know, but it'll keep you sane and focused later on when the real work begins.

A word on the choice of prefixes—unless the PCB's solder mask print designated the components using unconventional prefixes, it's better to stick to the standard ones given in Table 2.2 below:

Table 2.2 Components and their prefixes⁵²

Components	Prefixes	Components	Prefixes
Battery	B, BT	Integrated Circuit	U, IC
Capacitor	C	Jumper, Link	JP
Capacitor (decoupling)	CU	Potentiometer	RV
Connector (Jack)	J	Power Supply	PS
Connector (Plug)	P	Relay	K, RL
Crystal Oscillator	Y	Resistor	R
Delay Line	DL	Resistor Network	RN
Diode	D, CR	Switch	S
Display, LED	DS	Test Point	TP
Filter	FL	Transformer	T
Fuse	F	Transistor, MOSFET	Q
Hybrid Device	HY	Zener Diode	Z, VR
Inductor	L		

You will also notice that in my sample BOM (Table 2.1) there is no standard microcircuit drawing (SMD) numbers (ICs beginning with M38510), though there are discrete parts that do (capacitors beginning with M39003 and M39014). Why is that so?

When I first started out, I did try to be consistent by including the part numbers in the BOM as they appeared on all components, regardless of discrete or integrated. However, I soon realized that as far as the ICs are concerned these long, archaic alphanumeric entities did not serve any meaningful or useful purpose, only frustrations as I had to constantly cross-reference their commercial equivalent to know what they are. In the end I dropped the idea and just stick with the commercial part numbers for ICs, while retaining the SMD numbers for discrete capacitors, since they are just capacitors and I do not really have to bother about their values and ratings, unless I want to include them in the schematics and only when I have the luxury and time to do so anyway.

Remember, a BOM is only useful so long as it helps you identify the components on the PCB. You can spend a large part of your time to include as much information as you want, but make sure these are the information you really need that can speed up your work later on.

⁵² Some people prefer a longer prefix because it's more readable, such as RNET for resistor network, or JMPR for jumper. My advice is to keep it as simple as possible to just one or two letters. It will prevent clutter when you do PCB layout and name the components later on.

CONFORMAL COATINGS

As the term implies, a conformal coating is a thin film (25-75µm)⁵³ of protective chemical substance (or membrane) that coats over a PCB or electronic assembly by conforming to its contours and components. Conformal coating in effect applies a layer of insulation on the PCB against moisture, dust, heat, fungus, and corrosion, etc.

Not all PCBs are conformal coated—motherboards found on personal computers and laptops usually aren't, including circuit boards that are installed in most mobile phones.⁵⁴ In fact, many PCBs found in commercial household products like TV, washing machines, fridges, and hi-fi sound systems seldom are, except in countries where the weather and environment warrant its use. Military circuit boards and PCBs used in aeronautical designs, on the other hand, because of the harsh and often extreme conditions in which they operate in, must of necessity be conformal coated to ensure they continue to be functional and reliable.

PCBs that are conformal coated usually have a glossy shine⁵⁵ on both the component and solder sides, and will exhibit a luminous blue glow when placed under ultraviolet (UV) lighting. Table 2.3 lists five types of material you will most probably encounter:

Table 2.3 Types of Conformal Coatings⁵⁶

Material	Properties			Rework	Usage
	Surface Adhesion	Chemical Resistance	Humidity Resistance		
Acrylic	Acceptable	Poor	High	Easy	High
Epoxy	Good	Excellent	Acceptable	Difficult	Seldom
Paralyene	Excellent	Excellent	Excellent	Impossible	Rarely
Urethane	Good	High	Acceptable	Difficult	High
Silicone	Poor	Low	Excellent	Possible	Moderate

Most conformal coated PCBs I worked on so far use some form of acrylic resin as the medium, which is quite easily removed with solvents like the Humiseal 1080, a VOC-compliant⁵⁷, non-ozone depleting chemical.

⁵³ Certain kind of conformal coatings may be as thick as 200µm, such as silicone, for example.

⁵⁴ It might surprise some of you to know that the PCBs that power modern mobile phones are actually a scaled down version of embedded computer systems, some boasting multi-core processors like their PC cousins!

⁵⁵ The exception is a PCB with non-glossy Paralyene coating in which it's so thin it's almost invisible under normal lighting.

⁵⁶ Military specification MIL-1-46058 recognizes these five types of conformal coatings.

⁵⁷ VOC stands for volatile organic compound, and refers to chemical with a high vapor pressure at normal room temperature. It is essential to have a separate coating room that is properly ventilated, and you should also put on a good respiratory mask when performing conformal coating removal or re-application. Health matters!

There are rare instances where the coating is epoxy or polyurethane, in which case I resorted to running a fine-pitch file gently over the tips of the component legs to lightly scrap off the coating to expose the conductive solder,⁵⁸ and then brushing off the flakes using an anti-static brush.

Another thing about removing conformal coating is deciding whether to strip only the solder side, or both the component and solder sides. This is applicable if the PCB is through-hole since the component legs can be accessed from the solder side. Stripping just the solder side has the advantage you only need to re-coat that side after you're done with your job. The flip-side is it's a tedious process because you have to do it manually by holding the PCB with one hand and using a brush to constantly dip into the solvent tray and applying the solvent on one half of the PCB, taking care not to let the solvent flow over to the component side,⁵⁹ and then repeating again on the other half of the PCB, while checking it under UV light to see if the coating has been completely removed.

Once the reverse engineering work is completed, it will be necessary to re-coat the PCB, in part or total, depending on how extensive the removal was in the first place. My preference is the Humiseal 1B31 can spray, a fast drying acrylic which is quick and easy to apply by layers for fully stripped PCBs. For partially stripped or component legs with coating scrapped off, I use cotton buds dipped in acrylic solution to lightly touch up the affected areas.

Personal Protection Equipment (PPE)

VOC chemicals used for conformal coating removal are toxic and hazardous if inhaled over extensive periods of time. Even the normal alcohol used for cleaning the PCBs after soldering work can get you high after a while and induce withdrawal syndromes. So do your health a favor and get a good face mask respirator like the ones shown below:

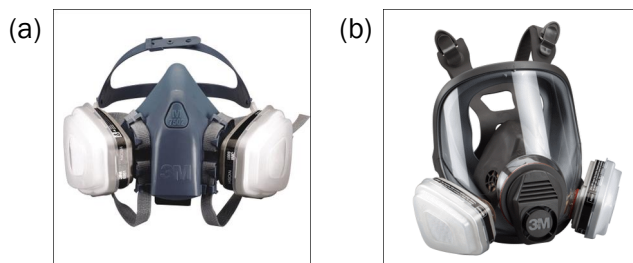


Figure 2.21 Types of PPE: (a) Half Face-piece face mask, and (b) Full Face-piece respirator. (Courtesy of 3M)

⁵⁸ Micro abrasive blasting may help with removal of tough coatings such as Paralyene, but care must be taken with the selection of abrasive materials and air pressure to prevent damage to the PCB. I would discourage this method with epoxy and polyurethane as it tends to leave stubborn residues that require extra cleaning.

⁵⁹ It will be impossible to prevent the solvent from flowing over to the component side of the PCB if there are unfilled via holes present. You need to decide if it's worth the trouble to strip only one side to save on the solvent and re-coating costs. If you intend to do the manual way, do take note of the precautionary measures mentioned in footnote 57 above!

COMPONENT DATASHEETS

We now come to the last part of the basic preparation work—collecting component datasheets. I cannot emphasize enough the importance of having the necessary information ready on hand as you work on the PCB. There is nothing more frustrating than getting stuck with a component and not knowing its pin-outs or functions to meaningfully relate its role to that part of the circuit it is located at. Doing PCB reverse engineering is more than just blindly tracing out the interconnectivity of the components on a PCB, it is the ability to make sense of the board's topology and organize the various pieces of the puzzle as they form, and finally to recreate the schematic diagram as close to the original design, and as usable (and readable) as possible.

In this respect, component datasheets serve a number of purposes:

- In terms of PCB layout, provide the physical dimensions of the package so you can:
 - a. Create the layout symbol⁶⁰ if it's not found in the Visio stencils,
 - b. Accurately represent the component to scale on the layout diagram.
- In terms of the PCB schematic:
 - a. Provide the component pin-outs to help you determine possible connecting points, and whether the traced out paths are logical and correct,
 - b. Provide the truth-table functions or signal properties (input, output, bidirectional, etc.) should the pin-outs are not obvious or self-explanatory in themselves,
 - c. Provide design application notes that give hints on how the component can be used in relation to other components.⁶¹

I will not go into the details of how to interpret the various sections of a datasheet; being an electronic engineer yourself, I assumed you should be quite capable to read and understand datasheets in general without much problem.⁶² Having said that, I will still make reference to the relevant sections of a data-sheet when the occasion requires it.

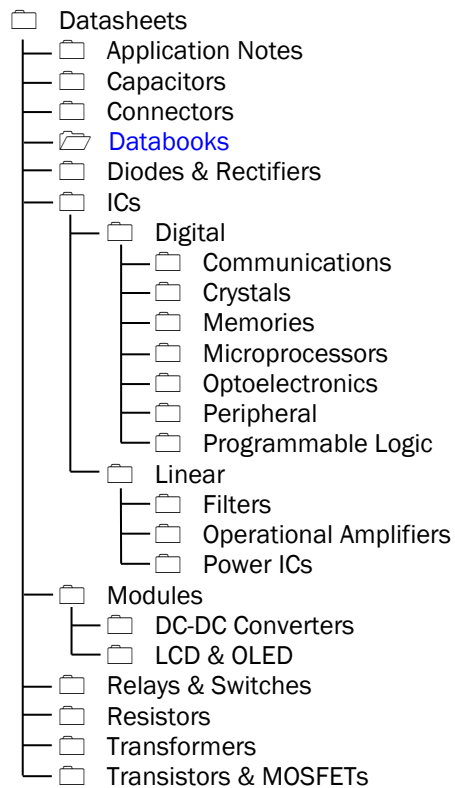
In the course of doing PCB reverse engineering, you will no doubt come across common components like the TTL and CMOS integrated circuits. Instead of searching each for these ICs individually and filling up your datasheet directories with myriads of these pdf files, I recommend that you download their complete databooks once and for all. For every project that I work on, I'll create a main directory under the same PCB name and several sub-directories, one of which will be the datasheets repository for components specific only to that PCB, but common components will not be included.

⁶⁰ Layout symbol is different from footprint symbol; the first usually refers to the top view of the component which includes the package's body and pins, while the latter has to do with the PCB pads on which the component legs are soldered onto. You can choose to create your own layout symbols if you do not like the ones Visio provided.

⁶¹ It's not surprising that some PCB designers follow the recommendations of design application notes verbatim since it is the safest bet that the example circuits will work as intended by the chip manufacturers. We will look at an example of this in [Chapter 4](#).

⁶² If you've been in the electronics trade for three years or more, chances are you would have already encounter some component datasheets already. For a formal treatment of this topic, you might want to read [Understanding and Interpreting Standard Logic Data Sheets](#) by Stephen M. Nolan and Jose M. Soltero, engineers from Texas Instruments. This article is available as a downloadable pdf file by searching its title online.

Instead, I created a separate DATASHEETS directory with the following sub-directory structure:



You can do the same and place all the databooks (TTL and CMOS from Texas Instruments, Motorola, and National Semiconductors, etc.) into the [Databooks](#) sub-directory for quick search and reference.

Whew! We're finally done with the tedious stuff. I hope you had the patience to scurry through the loads of information as much as I had gritted my teeth to write them. And I hope you're looking forward to poring through the next two chapters and working out the steps, because I'm roaring to start writing—after I take a much deserved break!

Chapter 2

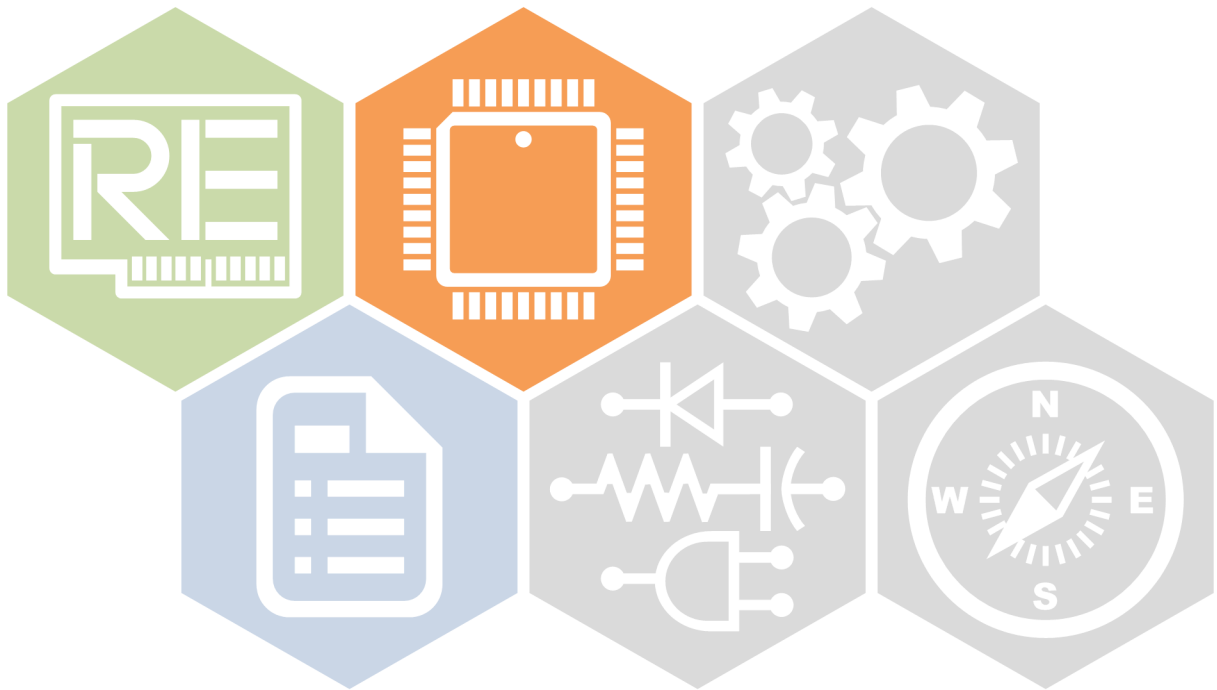
Two days later...

Alright, break's over!

3

Laying Out The PCB Artwork

**Learning a new skill
may not be as hard as you think...
you might just get it on the first try!**



CONTENT

Is it Really Necessary?–68

Getting to Know Microsoft® Visio–69

Initial Visio Settings–70

 User and Developer Modes–72 Rulers and Grids–73

 Guide Lines and Guide Points–73 Snap and Glue–74 Font Styles–74

 Text Styles–75 Line Styles–75 Fill Styles–75

Keyboard Shortcuts–76

Quick Maneuvering Tricks–76

 Zooming–76 Panning and Scrolling–77

Manipulating Objects–78

 Aligning Objects–78 Distributing Objects–78 Arraying Objects–78

 Sizing and Positioning Objects–79

A Simple Example–80

Preparing the Drawing Page–81

Drawing the PCB Outline–86

 Drawing the PCB Main Body–87 Drawing the Card Edge Connector–88

 Drawing the PC Slot Holder–89 Summarizing and Recap–92

Creating Component Layout Symbols–93

 Discrete Components–93 Integrated Circuits–97 Miscellaneous–101

Populating the PCB–102

A Complex Example–106

More About Circuit Boards–108

PCB-Mount Fixtures–109

Bolts and Nuts–111

Connectors–112

 96-way DIN Connector–112 68-way SCSI-2 Connector–113

 100-way I/O Connector–114

Toggle Switches–115

Complex IC Layout Symbols–116

 Ball-Grid Array–116 Pin-Grid Array–116 Small Outline ICs–117

 Quad Flat Package–118 Plastic Leaded Chip Carrier–118

 PLCC Footprint–119 PLCC Socket–119

Drawing Analog Components–121

 Types of Semiconductor Packages–122 Transistors and Heatsinks–122

 Diodes–125 Metal Can Analog ICs–126

Mobbed!–127

Too Small To Ignore–130

3

LAYING OUT THE PCB ARTWORK

"Everything should be made as simple as possible, but not simpler."

Albert Einstein, 1879-1955

The thought of creating a PCB layout may seem daunting. Consider the following example from my own portfolio:

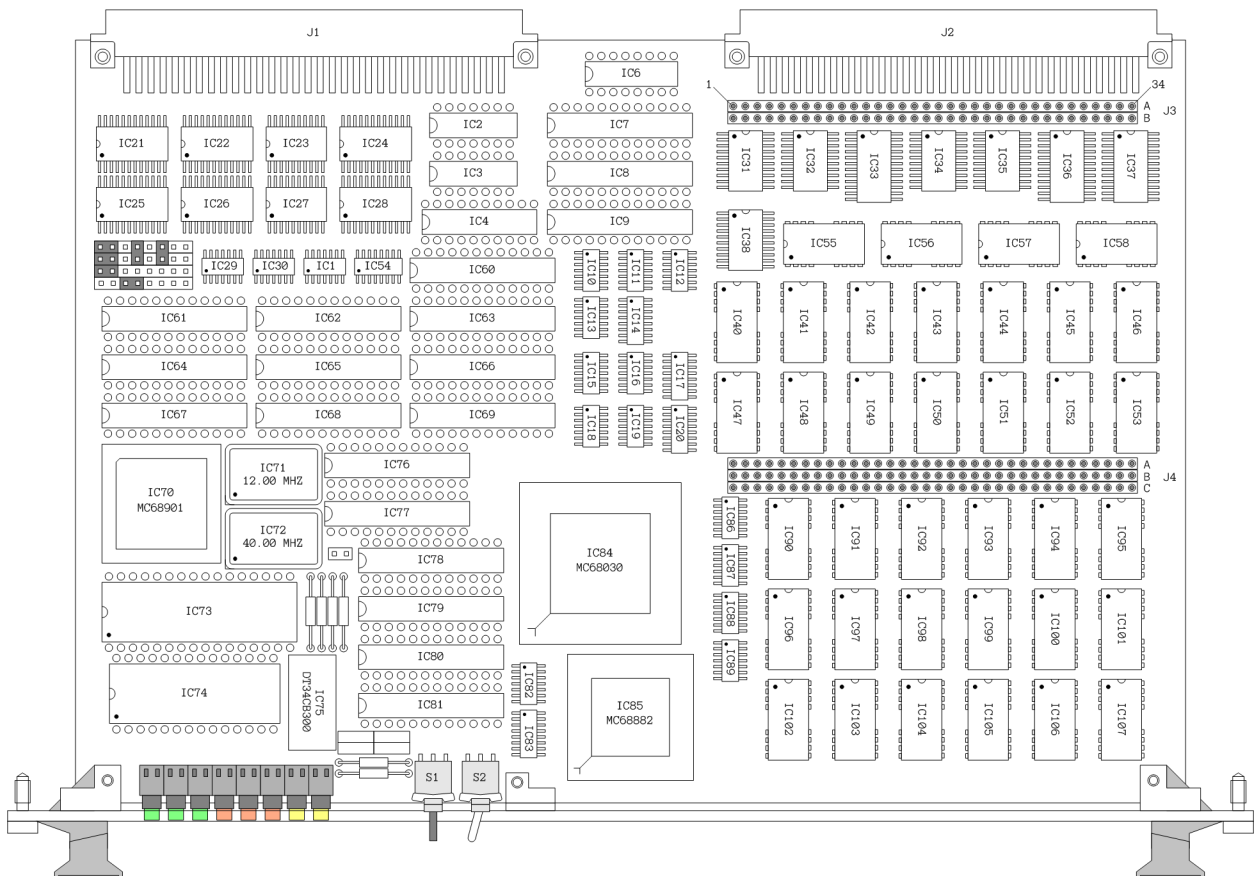


Figure 3.1 Sample PCB layout (component side).

Yeah, I know what you're thinking right now. Some of you might be thinking, "Wow! Is it possible for me to draw THAT?" Others might be contemplating returning the book to the store, "You gotta be kidding! I could never do what you did even if I try..." Hopefully there'll be some who'll respond, "I like what I'm seeing and I'm gonna give it my best shot and see what I can come up with!" And as Albert Einstein suggested, I'll make it as simple as possible. I promise.

IS IT REALLY NECESSARY?

Well, that depends on your style of working or preference. Some might think it's a hassle and prefer to just photocopy the PCB and use it as the basis to work on the schematics. I won't stop you from doing that, but allow me to give you a word of caution from my own experience—it'll get messy and confusing as you progress, and the less than ideal resolution of most photocopy printout might prove too much for your eyes, especially if the PCB is densely populated; also, if there are missing reference designators, you'll have to manually label them. And each time you screw up and need to redo the printout, it's going to be a perpetual pain you'd wish you had a decent layout diagram to print on demand.

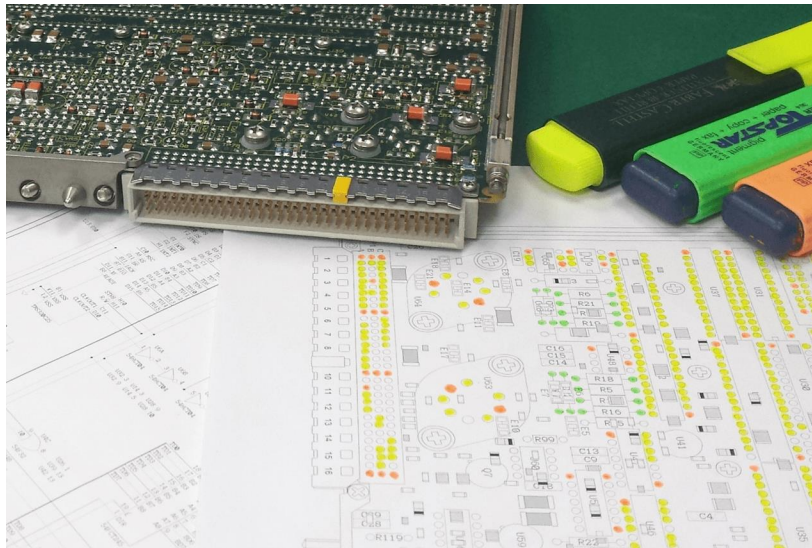


Figure 3.2 A PCB layout diagram used for highlighting probed points.

A PCB layout diagram serves the following purposes:

- Proper documentation of the PCB, including missing reference designators and additional data you might care to put in, such as the BOM side by side for quick reference.
- Ease of locating specific components since it is in electronic form and therefore searchable even across multiple pages.
- Facilitate marking (highlighting) of probed points to allow you to view your progress and cut down on repetitive probing, saving time and reducing wear or possible damage to the PCB.

Having some knowledge about the tool you'll be using to draw the PCB layout is essential, so...

GETTING TO KNOW MICROSOFT[®] VISIO⁶³

If you think Microsoft Visio is only good for doing flowcharts, organizational charts, or floor plans using its supplied stencils, I hope the earlier layout drawing will change your mind set. As far as I'm concerned, Visio does a pretty amazing job with 2D technical illustrations.⁶⁴

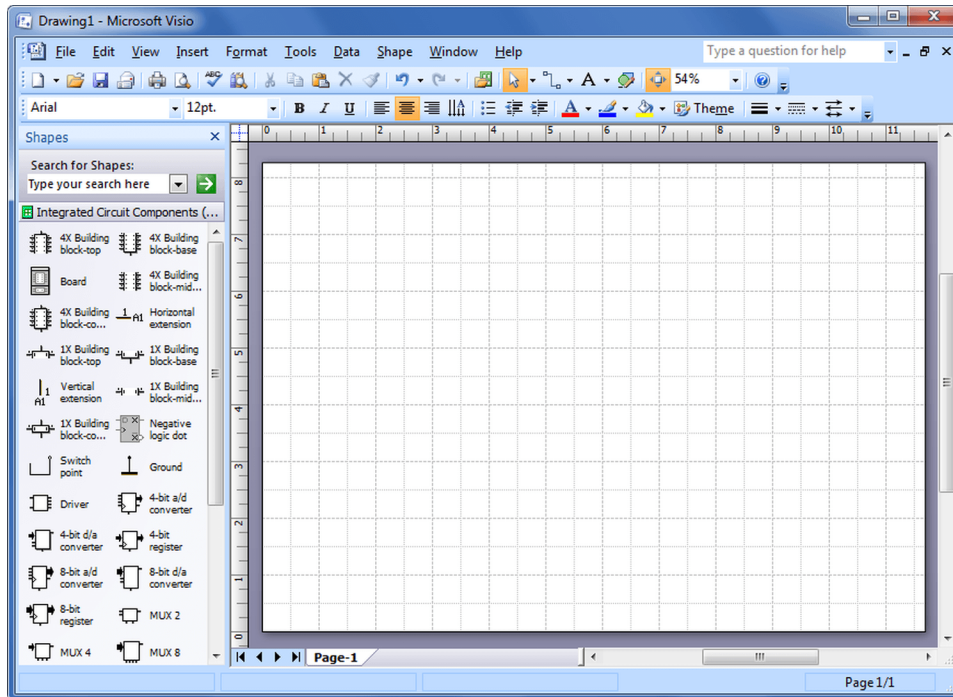


Figure 3.3 The Microsoft Visio 2007 workspace.

The Visio workspace is intuitively organized with the menu bar, standard and formatting toolbars on top, the shape stencils stacked to the left (they only appear if you open them), and the main drawing page occupying the bulk on the right. Less obvious are the page tabs located below the drawing area, usually beside the horizontal scroll bar (something like Excel sheet tabs), which is the standard layout for all Microsoft Visio versions. It's not really a big deal, though, since navigating between pages of a drawing is not difficult to figure out.

I won't go into a formal tutorial on how to use the features of Visio as that will take up too many pages, which is not what this book is intended for. If you have experience with Windows applications then it will not be a problem finding the functions when I mention them as I go along. However, just to be sure, I've included in [Appendix B](#) the Microsoft Visio 2007 Quick Reference Guide by CustomGuide which

⁶³ The version of Visio I'll be using throughout this book is Microsoft Visio 2007, though I've used Visio Technical 4.5 since 2001, until my company upgraded its corporate volume licensing Microsoft Office suite to 2007. Generally Visio's basic features and functions have not changed much, though overall aesthetic appearance has improved and the standard toolbars replaced with the ribbon toolbar from version 2010 onwards. Since buying over Visio, Microsoft has released Visio 2000, 2003, 2007, 2010 and 2013 for Windows.

⁶⁴ I've even used Visio to do some pseudo 3D drawings for that matter!

gives a summary of the common commands and shortcuts. If you're using a different version of Visio, you can go over to their website to download the relevant one.

INITIAL VISIO SETTINGS

When Visio first starts up, it will show the [Getting Started](#) page with the [Template Categories](#) column on the left, and some recent templates and documents listed, if any, so you can quickly select or resume your work.

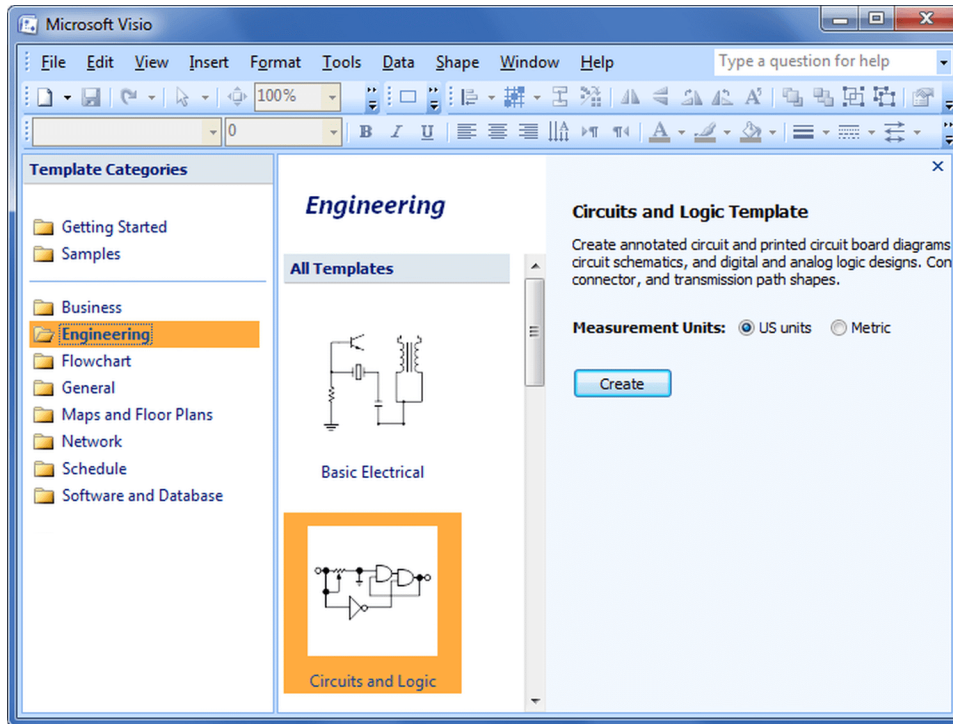


Figure 3.4 Visio start-up with engineering template category selected.

If you want a clean start, you can click on any of the Template Categories such as [Engineering](#), then select the [Circuits and Logic](#) template as an example, choose the measurement units and then click [Create](#). Visio will take you to the drawing page with the relevant shape stencils opened on the left.

A much faster way would be to double-click a Visio drawing file (.vsd) to instantly open up the drawing page with all the initial settings pre-configured. This is my usual preference as it saves me a lot of time and ensures that every Visio drawing I create is consistent in style and format. For this reason, I've created four standard Visio drawing files⁶⁵ as blank templates to choose from whenever I want to start a new project: a portrait and a landscape version each for both A3 and A4 sizes.

⁶⁵ As I mentioned in the [Preface](#) under [Additional Resources](#), you can obtain these blank Visio drawing templates along with some sample shape stencils from my download page if you buy my book.

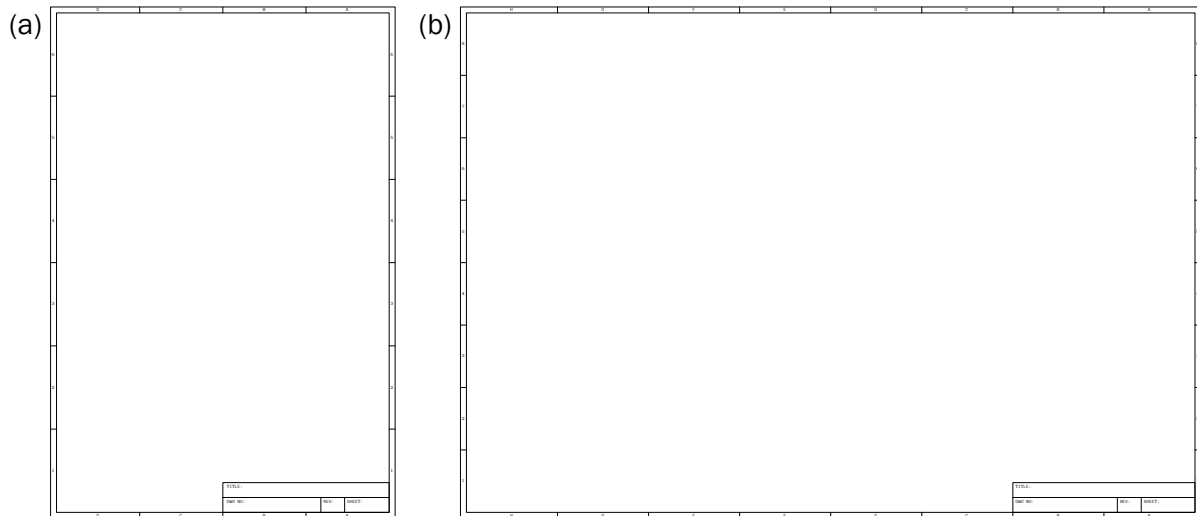
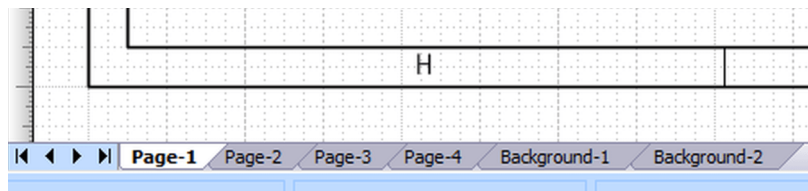


Figure 3.5 Blank template drawings with grid: (a) A4 portrait, and (b) A3 landscape, complete with a simple title block on the bottom right.

One of the features I like about Visio is it supports multiple pages per drawing file, even in the versions before Microsoft bought over, something which many other 2D diagramming software of its kind lacked. Best of all, you can designate a page to be either a drawing (foreground) page or a background page—and Visio supports multiple background pages too—which means you can have individual drawing pages each having a different background or no background at all. Neat huh?

In some of my drawings, I created two backgrounds with similar grids but different title blocks, one with the company logo and document details for the first page, and one minimally for the other pages.



While we're still on this topic of backgrounds—you can in fact create a master outline drawing of the PCB as the background page and link it to a few drawing pages, with each page showcasing different component parts (resistors on page 1, capacitors on page 2, diodes on page 3, etc.) to ease congestion and make for a clearer presentation. There is, however, a better workaround which will be discussed in [Chapter 5 - Advanced Topics](#).

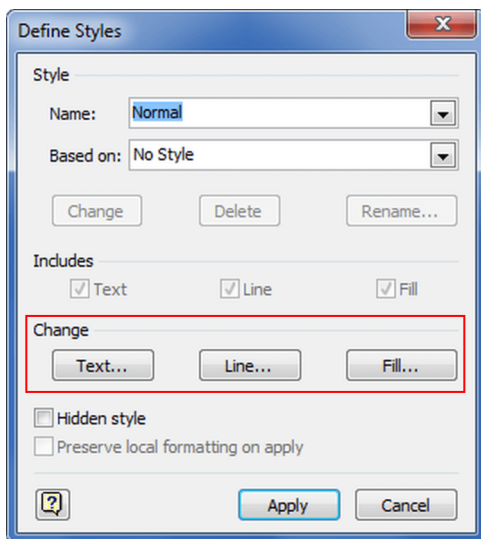
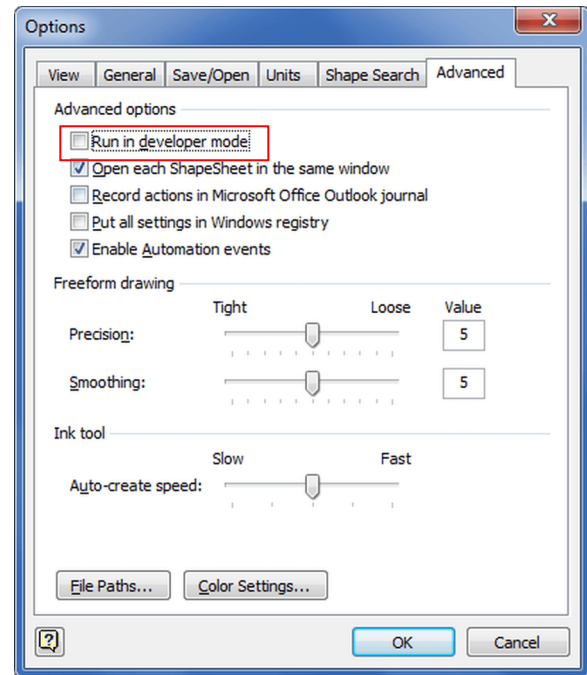
We will now talk about some initial settings before we start working in Visio. This involves changing the Visio's environment settings, as well as default formatting for font, shape fill, and line weight to make your drawings conform to the style and format you want and then saving it as a template, so you do not have to change or set them every time you create a new drawing by basing on that template's settings and styles.

User and Developer Modes

When you first install and run Visio you are by default in user mode, which means certain advanced options are not visible in the menu for you to access.

In order to effect the change of styles and formatting you'll need to switch to the developer mode. To do this, go to **Tools** → **Options...** and then click on the **Advanced** tab, under Advanced options select **Run in developer mode**. Now when you click on **Format** in the menu bar, you will see **Style...** and **Define Styles...** options available to you for setting the default styles for lines, shapes and font.

Note that there is a distinction between changing the formatting under **Define Styles...** and that of **Format** → **Text**, **Line** or **Fill...** or using the toolbar buttons. The former is a permanent change to that particular Visio drawing's formatting while the latter is temporary in nature and will revert back to the default styles it was created with.



To define your preferred default styles, go to **Format** → **Define Styles...** to bring up the dialog box. In the Style option, click the down arrow for **Name** and select **Normal**, the basic underlying style which most formatting are based on.

You can then proceed to change or set the text, line, and fill properties by clicking on their respective buttons under the **Change** option. However, we will not do that just yet until we come to the relevant part later on.

Many users are confused and frustrated in not being able to globally change these default formatting, as evidenced in many Visio discussion forums. I was a little disappointed at first but decided to just stick with templates as a workaround solution to this limitation which was imposed for whatever reasons.

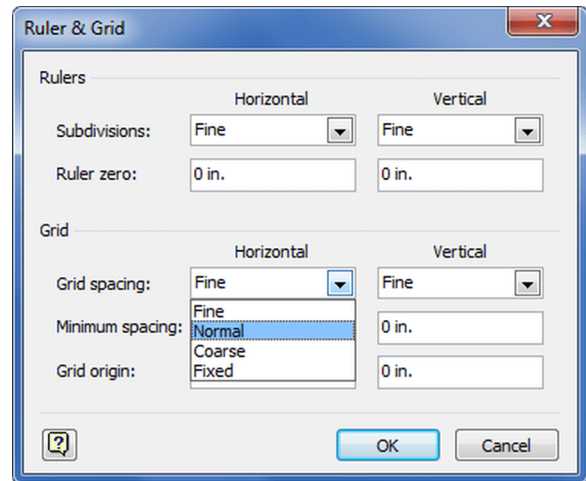
Let's move on...

To enable you to draw accurately and consistently, Visio provides a pair of rulers, grid lines on the drawing page, two types of guides, and the ability to snap and glue. You can change the ruler, grid, snap, and glue settings at any time without affecting the objects that you already placed in your drawing. The settings apply globally to all the pages in a drawing.

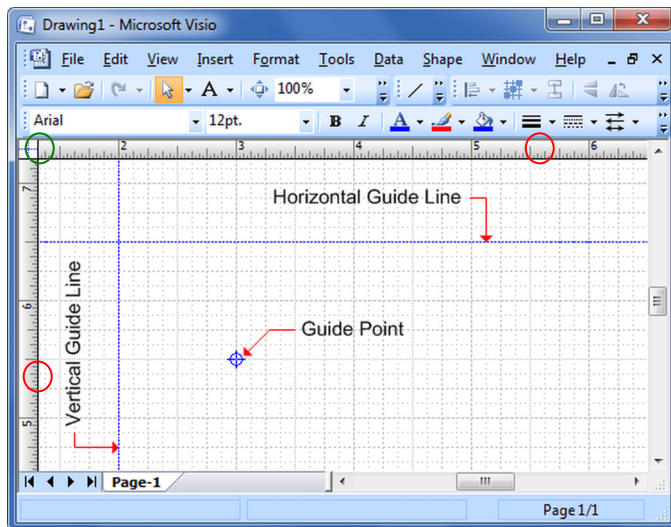
Rulers and Grids

By default, Visio sets both the horizontal and vertical rulers' subdivisions and grid spacing to **Fine**. The ruler is OK but the grids can be too fine for comfortable viewing, and may obscure drawing details especially if the PCB has many small SMD components.⁶⁶

Go to **Tools** → **Rulers & Grid...** to bring up the Ruler & Grid properties dialog box, and set the horizontal and vertical grid spacing to **Normal**. This will reduce screen clutter and allow you to see drawing details more clearly.



Guide Lines and Guide Points



Two very useful tools to help align objects on the drawing page are the guide lines and the guide points.

Guide lines are like movable grid lines which you create by dragging from the rulers (red circles), horizontal guide lines are made from the horizontal ruler, and vertical guide lines from the vertical ruler, respectively.

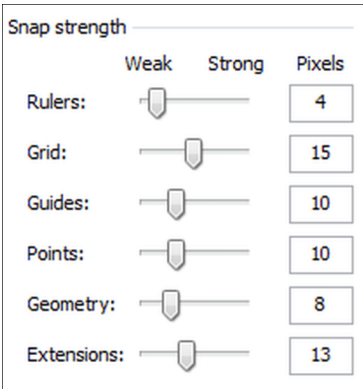
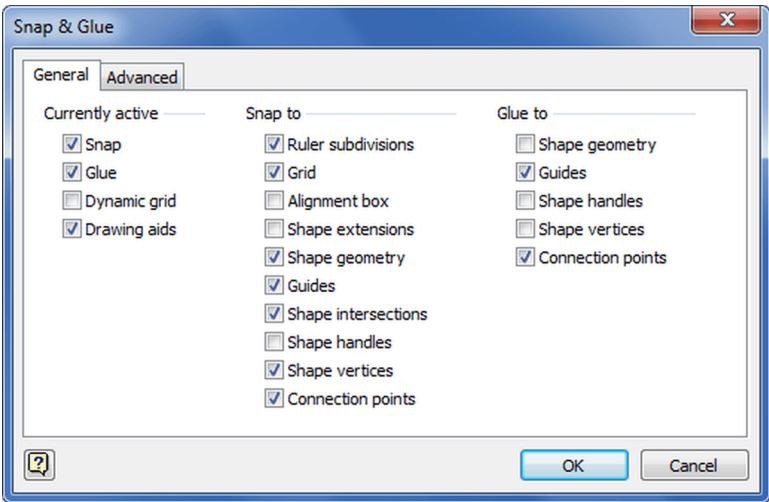
Guide points on the other hand are just circles marked with a plus (+) sign in the center, and are formed by dragging from the intersection of the rulers (green circle).

You can create as many guide lines and guide points as you need to help you align objects and shapes, but too many of these will likely clutter and obscure your drawing, so use them conservatively.

⁶⁶ It's really a matter of personal preference. To give you an idea, if your measurement units is in inches and your zoom is set to 100%, **Fine** will display 8 grid lines per inch, **Normal** displays 4, and **Coarse** displays 2. If you do not set the grid spacing to **Fixed**, the number of grid lines will increase as you zoom into a drawing. This can be quite disorientating for some users.

Snap and Glue

By default, Visio enables the snap and glue feature so you can create shapes or lines accurately by snapping them to different reference elements. My preference is to snap to grid lines so I usually set the snap strength for **Grid** to a higher value⁶⁷ than the rest. You can disable snap and glue and choose to draw freehand instead but I don't encourage that, unless you are prepared to struggle with the endless trimmings and adjustments that accompany freehand drawing.⁶⁸



Snap and glue options | snap strength settings for reference elements.

Font Styles

The choice of fonts does play an important part in how your drawings will turn out. Fonts are largely grouped into proportional or monospaced, with each serving different purposes in bringing out the best of a document, textual or illustration. Below is an example of each:

Proportional font: **ABCDEFGHIJKLMN OPQRSTUVWXYZ** **0123456789** (Arial 10 pts)
Monospaced font: **ABCDEFGHIJKLMN OPQRSTUVWXYZ** **0123456789** (Anonymous 10 pts)

Notice the 'slash' in the zero for the monospaced Anonymous font? It helps to distinguish a (0) from a (0). I have experimented with many such fonts and have come to like this particular one. In fact, it has become my preferred font for both PCB layouts and schematic diagrams.

For proportional font, I stick to the well-tested and popular Arial font for normal sentences and side notes. For titles I use its bigger cousin, the Arial Black font, for more weight and emphasis.

Before you start drawing, it's advisable to set the following properties to your preference so you don't have to keep changing them whenever you create a shape or text:

⁶⁷ The default value for Grid snap strength is 3 which is quite weak; I usually set it to 15.
⁶⁸ It would be neat if you could do freehand drawing or editing of shapes while the snap and glue is in effect by simply pressing the ALT key before drawing or editing a shape. This feature was only included in versions 2010 and higher, so if you really want this feature badly get the more recent Visio releases.

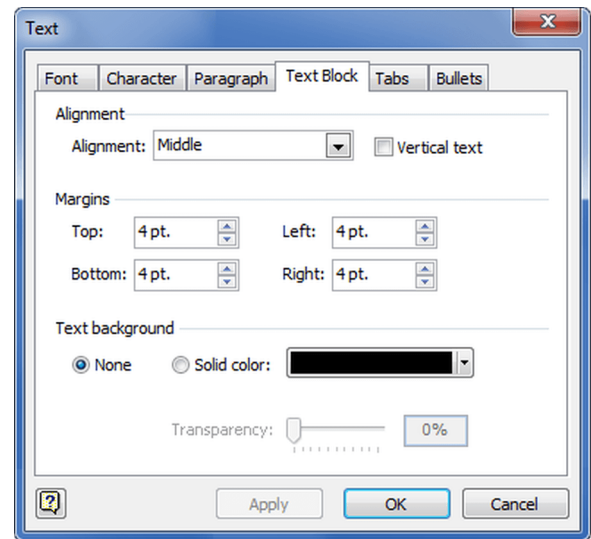
Text Styles

Go to **Format** → **Define Styles...** Make sure the Style is **Normal**, then click on the **Text...** button to bring up the Text dialog box. Change the following properties tab options:

Font Latin text: Anonymous⁶⁹
 Size: 6 pt.
 (Do not worry that 6 point is too small as Anonymous is a large font and this size is just right!)

Paragraph Alignment: Centered
 Spacing | Line: 120%

Text Block Margins | Top, Bottom, Left, Right: 0 pt.



Line Styles

Now click on the **Line...** button to bring up the Line dialog box. Change the following line options:

Pattern 01 – Continuous solid line
Weight 01 – Finest (approximately 0.25 pt. thick)

Fill Styles

Finally, click on the **Fill...** button to bring up the Fill dialog box. Change the following fill options:

Pattern 00 – None

That's it! Click **Apply** and the changes will take effect. Save your work now. When you open this Visio file again, the styles and format will be just as you've set them.

⁶⁹ This font is not included with Microsoft Windows so you'll need to go online to search, download, and install it to make it available for your use. As far as I know, this font is free under the Open Font License (OFL) which includes permission for commercial use. Refer to <http://scripts.sil.org/OFL> for more details. Credit and thanks to Mark Simonson for his generosity!

KEYBOARD SHORTCUTS

Event-driven applications running in Microsoft Windows offer users many ways of accomplishing the same tasks. While beginners usually navigate the menu bar and formatting toolbars to get what they want, more experienced users tend to favor keyboard shortcuts since it's much faster and allow them to focus on the real work.

[Appendix B](#) Visio Quick Reference provides a list of many useful keyboard shortcuts but I want to focus your attention on the following twelve:

Bring to Front	CTRL + SHIFT + F	Group	CTRL + G
Send to Back	CTRL + SHIFT + B	Ungroup	CTRL + SHIFT + U
Cut	CTRL + X	Flip Horizontal	CTRL + H
Copy	CTRL + C	Flip Vertical	CTRL + J
Paste	CTRL + V	Rotate Left	CTRL + L
Undo	CTRL + Z	Rotate Right	CTRL + R

It'll be nice to just use CTRL alone instead of the combination of CTRL+SHIFT for three of the keyboard shortcuts above, but I guess Microsoft just want to be consistent across their office products.⁷⁰ You can try to memorize these shortcuts upfront, which should be quite easy since some are already familiar and similar to other Microsoft's office products. Rest assured that as you start working in Visio and use them often, it'll become second nature in no time.

QUICK MANEUVERING TRICKS

Moving around effortlessly while constantly defining the workspace presentation of the drawing page to achieve the best views and right zoom ratios are important maneuvers that enable you to be effective and productive with Visio.

Zooming

At first glance, Visio seems to limit to a maximum zoom level of 400% from its standard toolbar's drop down zoom menu. You can, however, set the zoom level with the Zoom dialogue box by going to [View](#) → [Zoom](#) → [Zoom...](#) and typing a number from 1 to 3098⁷¹ in the [Percentage](#) data entry box.

There is a faster way to zoom but there's a catch—you need to have a scroll wheel mouse, the one with a wheel like object between the left and right mouse buttons. Simply move the mouse pointer to the area of the drawing page you want to zoom in, hold down the CTRL key, then scroll the wheel one notch forward, and voila! You just zoomed in one level (or 2x) into that region you pointed at.

⁷⁰ For example, CTRL+U = underline, though I can't understand why anyone would want to underline text in a diagramming tool when they can draw lines.

⁷¹ Don't ask me why Microsoft decided on such a weird zoom percentage number instead of a round figure like 3000 or 3500. I'm quite happy that this is almost double what Visio Technical 4.1 could achieve.

To zoom out, just scroll the mouse wheel one notch backward while you hold down the CTRL key. It would be good to set the zoom at 100% before doing the scroll-wheel zoom so you get zoom levels that are easier to work with (200%, 400%, 600%, etc.) in increments of 200 onwards.

Panning and Scrolling

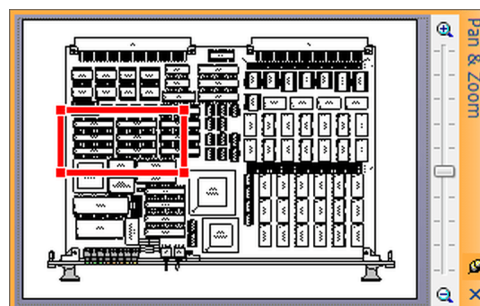
Panning is the ability to slide the drawing page horizontally from left to right and vice versa; scrolling has to do with the vertical up and down direction. Again, you'll need a scroll-wheel mouse to do this (so get one if you have not already done so!).

To scroll the drawing page up, simply scroll the mouse wheel forward; to scroll down the page, just scroll the wheel backwards. To pan the drawing page to the left, hold down the SHIFT key and then scroll the mouse wheel forward; to pan to the right, keep the SHIFT key depressed and scroll the mouse wheel backwards.

Most of the time as I work, I keep the last two fingers of my left hand over the left CTRL and SHIFT keys respectively, while I move the mouse pointer around with my right hand (unless I'm nibbling on a snack, of course!).

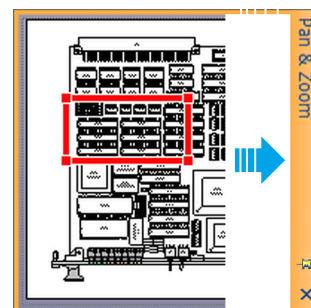
Question: Can you zoom, pan and scroll quickly without the use of a scroll-wheel mouse?

The answer is yes. Visio provides a tool known as [Pan & Zoom Window](#) (see right), which you can bring up by going to [View](#) → [Pan & Zoom Window](#). This is a small window containing a scaled down version of the main drawing page and a movable **red** rectangle corresponding to the zoomed area, which you can drag around with the mouse. There is also a zoom level control on the right side, with the minimum zoom set at page view depicted by the screen size.



The flip side is this window takes up some real estate on the drawing page, and even though you can enable the auto-hide feature, it still occupies an area equal to its title bar when hidden, and obscures a larger area when extended as you move the mouse pointer over it. This can be annoying for some users, and to me is more of a novelty tool.⁷²

One last tip: You can zoom and center the whole drawing page to the screen size by using the keyboard shortcut [CTRL+W](#) any time to re-orientate or get an overall view of your progress.



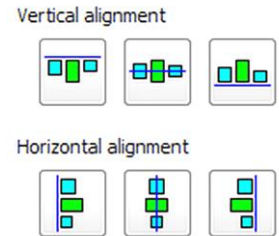
⁷² Navigation control windows of this nature is really helpful if your drawing page is A2 or larger, but I don't see any reason a PCB layout or even a schematic diagram should reach such proportions, though I've come across A1-size PCB assembly and schematic drawings in my work (which in my opinion is a waste of space, unless the designer happened to be suffering from severe presbyopia).

MANIPULATING OBJECTS⁷³

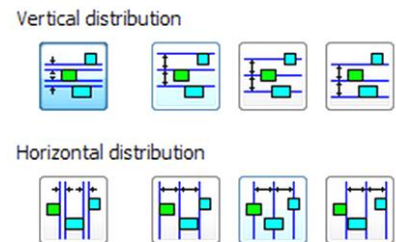
Instead of individually moving or lining up objects on a drawing page, Visio has the ability to manipulate these objects as a group when selected.

Aligning Objects

Once you selected a group of objects, the [Align Shapes](#) button on the Action Toolbar will become active and allow you to perform vertical (top, middle, bottom) or horizontal (left, center, right) alignments. Alternatively, you can go to [Shape](#) → [Align Shapes...](#) after you've selected the objects, though the first method is still the fastest and preferred.

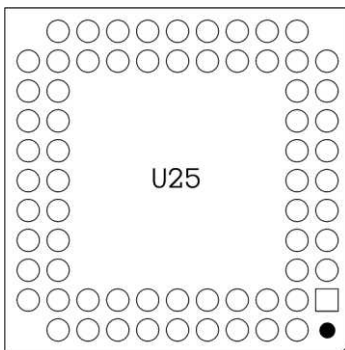
Distributing Objects

To evenly distribute or space objects apart vertically (top, middle, bottom) or horizontally (left, center, right), you can either click on the [Distribute Shapes](#) button on the Action Toolbar, or go to [Shape](#) → [Distribute Shapes...](#) after you've selected the objects.



Arraying Objects

There is a very powerful function stacked away under [Tools](#) → [Add-Ons](#) → [Visio Extras](#) known as [Array Shapes](#). I cannot understand why Microsoft chose to hide this nifty little gem in such an obscure level within the [Tools](#) menu.



Since discovering this function in Visio Technical 4.1, I have been using it a great deal to create many complex component objects for my PCB layout artworks, such as the PGA footprint shown on the left. Imagine drawing all the 68 pins of the IC individually and trying to align and distribute them even with the help of the afore-mentioned functions! It'll not only be a pain in the neck, but a strain on the eyes—and I mean it literally!

I will discuss more about this function later on when we learn to create component objects requiring array of shapes.

⁷³ While Microsoft uses the term [shapes](#) for the entities found on a drawing page, I prefer to call those entities which are made up of two or more shapes (basic primitives) [objects](#). In fact, a complex object can even be made up of many other objects of varying complexities, just like the universe is made up of galaxies from dwarf (comprising millions of stars) to super-clusters (formed by a group of galaxies with stars numbering in the billions)!

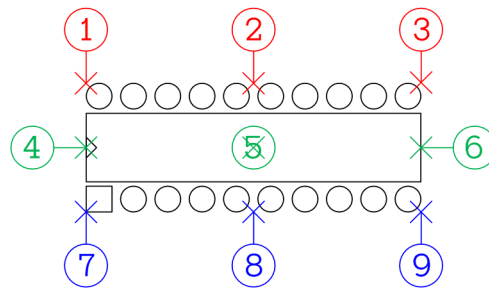
Sizing and Positioning Objects

Being able to define the size and placement of an object is also essential to creating components that best approximate the actual dimensions specified in their datasheets, ensuring that the PCB layout is accurately to scale and aesthetically appealing. This is accomplished with the Size & Position window which can be summoned by going to [View](#) → [Size & Position window](#).

X	4 in.	Size & Position
Y	6 in.	
Width	0.5 in.	
Height	0.5 in.	
Angle	0 deg.	
Pin Pos	Center-Center	
		X

Notice that besides the X-Y positions relative to the drawing page's origin, and the width and height of the object, you can also change its angle of inclination (positive for clockwise, negative for counter-clockwise) and the pin position by which the rotation or transformation is achieved.

Of interest is the object's pin position property, which defines the reference point for the object's X-Y positions, as well as the pivot point on which the object rotates. There are nine options to this property, namely: **top-left**, **top-center**, **top-right**, **center-left**, **center-center**, **center-right**, **bottom-left**, **bottom-center**, and **bottom-right**. Thus, a 20-pin DIP IC will have the following pin positions which you can set as its reference and pivot points:



It may not be obvious how this property can make your job easier, but for now, I would suggest that you keep the Size & Position window open as long as you're creating component objects, never mind that it takes up a little screen space. Just dock it somewhere convenient on either side of your drawing page with the auto-hide option enabled.⁷⁴

Now that we've covered the necessary basics and essentials of Visio, it's time to get down to business and do some serious but interesting work (Hooray!). First, I will go through the process of drawing the PCB layout using a simple example, and thereafter we'll do a more complex board.

Sounds good? Great! Let's get started then...

⁷⁴ In Visio Technical 4.1 you need to right click on an object and select the Size and Position option to bring up a dialog box, enter the necessary data and click OK. The dialog box closes and the object gets updated. With the Size & Position window, you simply click on an object and its current properties immediately shows, allowing you to make changes on the go easily. With each release of Microsoft's products, there are certainly new improvements that will make your work easier, though not always.

A SIMPLE EXAMPLE

Our first candidate is an ISA-bus NCR SCSI host adapter which I happened to find lying among some old PC parts inside my work desk drawer:



Figure 3.6 SCSI host adapter

This is a pretty simple through-hole PCB by today's standard, comprising just nine ICs, a DIP switch and some discrete components, housed in a slightly smaller than half-size 8-bit ISA board, complete with a female Centronics type SCSI connector.⁷⁵ Best of all, it has no conformal coating to remove. Possibly the most complicated IC will have to be the 40-pin NCR-5380 SCSI Interface Chip but it should not pose any problem as far as PCB layout is concerned. Here is the complete parts list:

U1	SN74LS688N	U9	SN74LS688N	C1	100nF	C9	100nF
U2	DIP Switch	U10	74LS244N	C2	100nF	C10	100nF
U3	SN74LS04N	R1	1000	C3	12.6nF, 6V	C11	100nF
U4	DM74LS138N	RP1	007-6718203	C4	100nF	C12	47uF, 6V
U5	SP8924	RP2	007-6718203	C5	100nF	C13	47uF, 6V
U6	SP8918	RP3	007-6718203	C6	100nF		
U7	NCR-5380	F1	1.5A, 125V	C7	100nF		
U8	D2764A	CR1	1N5817	C8	100nF		

⁷⁵ A number of these terms may seem foreign to some of my younger readers, and rightly so because they belong to a past era in computer history. ISA stands for Industry Standard Architecture (well, not anymore) which was the standard bus for first generation IBM-PC or PC-XT. The name Centronics is synonymous to the parallel interface connector used by printer models of bygone years. Fortunately, SCSI or Small Computer System Interface is still very much alive and in use today, having evolved to the parallel Ultra-640 protocol and even the 16GFC fiber-optics channel with a throughput reaching 12.8 Gbps!

Below are the specific steps to create the PCB layout:

1. Prepare the drawing page
2. Draw the PCB outline
3. Create the component layout symbols
4. Populate the PCB

PREPARING THE DRAWING PAGE

When considering the drawing page, you should first decide on which paper size standard you want to follow—metric or imperial:

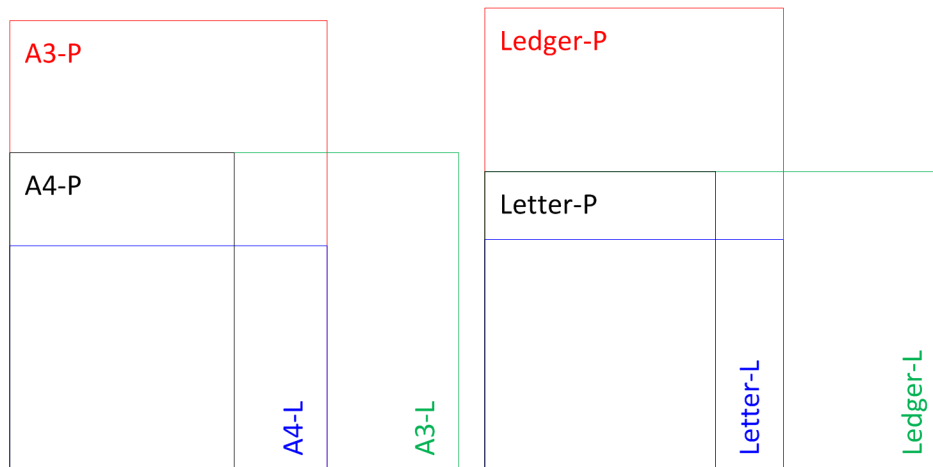


Figure 3.7 Page sizes and orientations (portrait and landscape) for metric ISO and US imperial paper sizes.

Most home printers usually accept the ISO A4 (210mm x 297mm) and imperial letter (8.5" x 11") as the standard paper sizes, while office printers can handle the bigger ISO A3 (297mm x 420mm) and imperial ledger (11" x 17") sizes.⁷⁶ This does not mean that you can't use A3 or ledger for your drawing page if you've only got a home printer with A4 or letter paper size capacity. You can and should use the paper sizes that meet your requirement.

I use the ISO A3 in landscape orientation quite often out of necessity and usually print my draft artworks using the [Fit to Page](#) scaling option that is available in many small laser printers. Though scaling down from A3 to A4 will reduce text and labels on the drawing by half, they are still readable to a person of my age with some degree of presbyopia.

In our case, the half-size ISA board measuring some 5-inch square should fit nicely in an A4 or letter size drawing page with room to spare, whether portrait or landscape. I'll use the landscape orientation for our illustration.

⁷⁶ Notice that ISO uses the millimeter while imperial uses the inch measurement unit. Most people will stick to the system their countries adopt and which they're most familiar with; for me, although we use the British metric system (Singapore was once a British colony), I still prefer the imperial measurement unit despite having to work with ISO A4 and A3 paper sizes. Call me an odd ball if you like!

To create a new drawing page, follow these steps:

1. Run Microsoft Visio.
2. Click on the **New** icon on the toolbar | press CTRL + N | **File** → **New** → **New Drawing** (US units)⁷⁷
3. **File** → **Print Setup...** and change the Printer paper option to **A4** and **Landscape**. Click OK.

You now have a blank drawing (or foreground) page named Page-1 to work on. We will create a background page to house the grid references and title block as shown below:

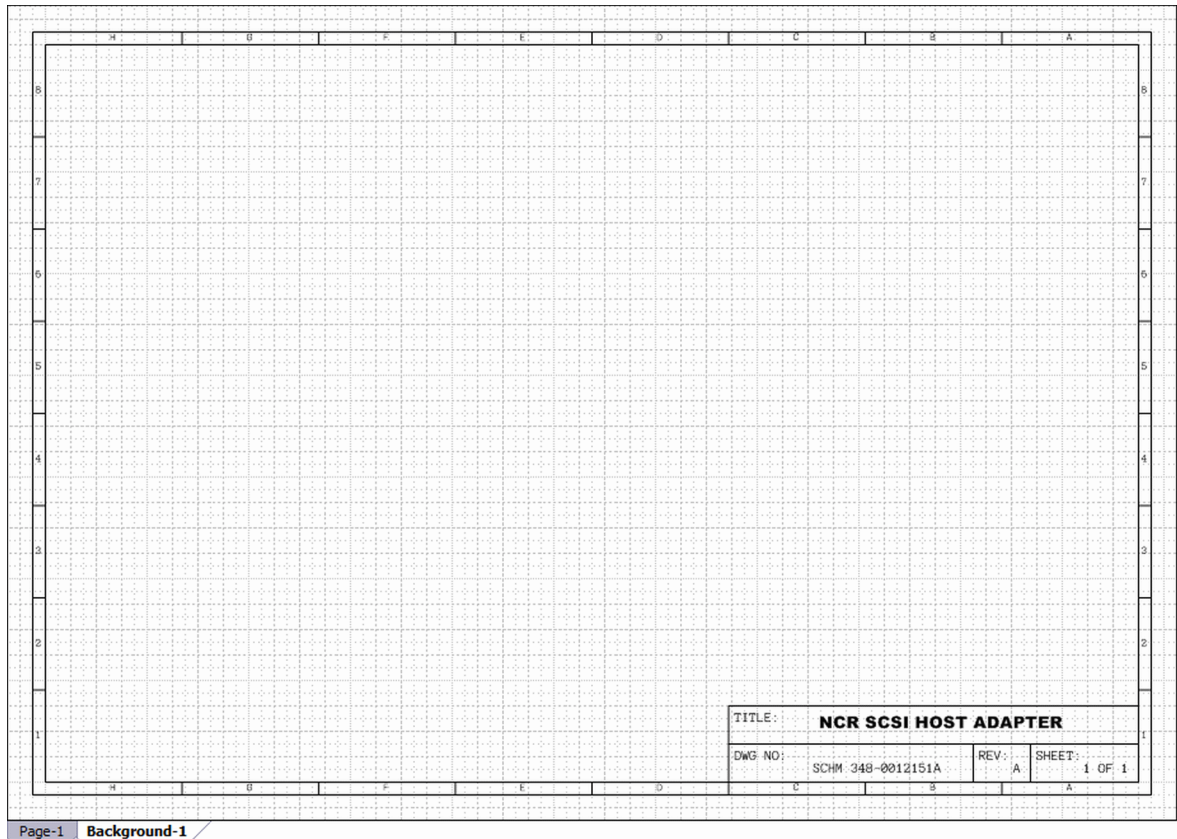


Figure 3.8 Background page with grid references and title block.

4. **Insert** → **New Page...** to bring up the Page Setup's Page Properties tab and select **Background** for Type. The Name box changes from Page-2 to Background-1. Click OK.

Notice that the page tabs region now has Page-1 and Background-1 tabs with the latter highlighted as the active page. Before continuing, make sure you have the initial Visio settings taken care of as discussed in the beginning of this chapter.⁷⁸

⁷⁷ Choose either one of the three options segregated by the '|' to achieve the same result.

⁷⁸ This involves setting the line weight to **1**, the fill option to **No Fill**, and the font to **Anonymous 6 pt**, etc. so that

We will now proceed to draw the grid reference:

5. Click the **Drawing** icon  on the toolbar to show the Drawing tools .
6. Select the Rectangle Tool or press CTRL+8.
7. With the drawing page in full **Page** view, draw a rectangular box from X=0.375, Y=0.375 to X=11.375, Y=7.875 (all numbers are in inches).
8. Draw a second rectangular box from X=0.500, Y=0.500 to X=11.250, Y=7.750.

Next we'll add the dividers (grids) and their references. You can draw one set of X and Y grids at the top and left regions, then duplicate them to the bottom and right.

Note: Make sure that the **Size & Position** window is opened for entering the X and Y location values in the following steps.

9. Create vertical and horizontal lines of length 0.125 such that they fit into the space between the two rectangular boxes (refer to Figure 3.8). For the vertical lines, position them at the following X locations: 1.8438, 3.1875, 4.5313, 5.875, 7.2188, 8.5625, and 9.9063. For horizontal lines, their Y locations are: 1.4063, 2.3125, 3.2188, 4.125, 5.0313, 5.9375, and 6.8438.
10. Now select the Text tool  and create a text object on the drawing page with the letter 'A'. Then resize it to 0.125 square. Next right click and select Format → Text... to bring up the **Text** properties dialogue box. Click the **Text Block** tab and change the margins for Top, Bottom, Left and Right to 0 pt.
11. Duplicate 15 copies of the text object and change the text to B, C, D, E, F, G, H for the horizontal grid, and 1, 2, 3, 4, 5, 6, 7, 8 for the vertical grid. For letters A-G, place them at the following X locations within the top bar lines: 10.5781, 9.2344, 7.8906, 6.5469, 5.2031, 3.8594, 2.5156, and 1.1719. For numbers 1-8, place them at the following Y locations within the left side bar lines: 0.9531, 1.8594, 2.7656, 3.6719, 4.5781, 5.4844, 6.3906, and 7.2969.
12. To create the horizontal grid references for the bottom bar lines, use the mouse cursor to click and drag over the A-G area, including the vertical lines, on the top bar to select them. Press CTRL+C to copy, CTRL+V to paste, and CTRL+G to group the objects. Then click and drag them into the bottom bar lines—do not worry if they are not aligned to the top grid references.
13. Now click on the inner rectangular box to select it, then hold down the SHIFT key and click on the object group created in step 12 above. Next click the down arrow beside the **Align Shape** icon and select **Align Center**. The object group will align to the top grid references. 
14. Repeat steps 12 and 13 to create the vertical grid references for the right side bar lines, but this time when aligning the object group to the inner rectangular box, select **Align Middle** instead.

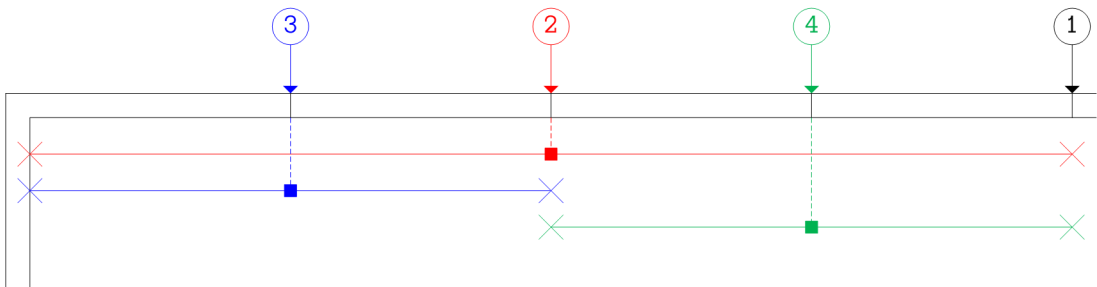
subsequent shapes are drawn at the desired line weight without color fill, and text or labels are of the correct monospaced font. Got it?

Up to this point, we have created a background with the grid references. Some of you might wonder if there is a more efficient way of aligning those horizontal and vertical lines, letters and numbers in steps 9 and 11, rather than keying in those decimal number locations. There is. But I've opted to do the idiot-proof way for two reasons: Firstly, it's not easy to describe the process upfront without confusing the less initiated of my readers; and secondly, I wanted to show just how useful and easy the Size & Position window enables you to place objects at specific locations.

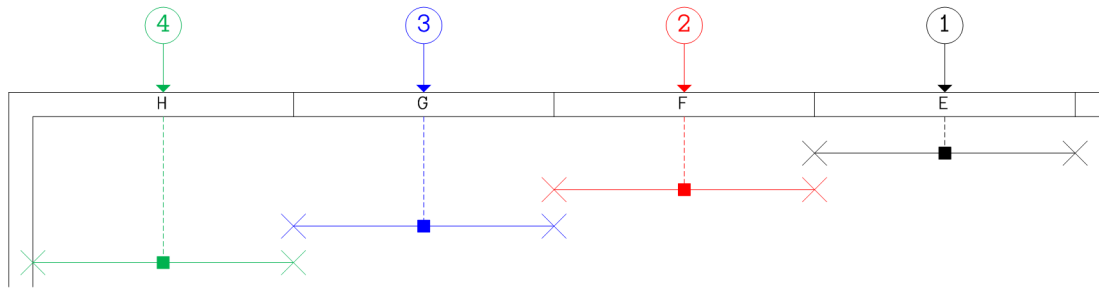
Now that I've achieved my objectives and in the process demonstrated the use of the Align Shapes tool, it's much easier to introduce the alternate method here, which I aptly coined the [Line-of-Reference](#) trick.

Beginning at step 9, we will create the vertical and horizontal lines this way:

- a. Create the first 0.125" grid line ① and place it in the top space bar. Next, select the inner rectangular box first, then hold down the SHIFT key and click the grid line ①. Use the Align Shapes tool to do an [Align Center](#) on the two selected shapes. Grid line ① is now centered with respect to the inner rectangular box.
- b. Draw a horizontal line (red) from the left edge of the inner rectangular box to grid line ①. Create a second grid line ② by either drawing or copying from grid line ①. Next, select the horizontal line first, then hold down the SHIFT key and click the grid line ②. Use the Align Shapes tool to do an [Align Center](#) on the two selected lines.
- c. Drag and shorten the horizontal line to where grid line ② is so it becomes half of its former length (blue). Create the third grid line ③ and repeat step (b) above.
- d. Shift the horizontal line (green) to align with grid lines ① and ②. Create the fourth grid line ④ and repeat step (b) above.



- e. Now that half the top horizontal grid is completed, you can repeat the other right half using the same method above, or simply copy grid lines 1-4, group them (CTRL+G), [Align Left](#) with respect to grid line ①, ungroup them (CTRL+SHIFT+U), and then delete the overlapping extra grid line at grid line ①.
- f. Doing the grid references is quite similar except that this time you only need to use the same horizontal line with a length spanning between two grid lines as shown below:



- g. Once you're done with the top bar, you can duplicate it for the bottom bar, then proceed to create the left and right side bars using a vertical line-of-reference, and repeat the steps above but using [Align Middle](#) instead.

By now, I hope you're getting better acquainted with the various ways of working in Visio, and even finding new ways that suit your style. We're left with the title block for the background page and having come this far, it should not be a problem for you to create one. However, to make this section complete, here are the details:

TITLE: NCR SCSI HOST ADAPTER		
DWG NO: SCHM 348-0012151A	REV: A	SHEET: 1 OF 1

Rectangle box	: 4.0313 (W) x 0.75 (H)
Dividing lines	: 4.0313 (horizontal), 0.375 (vertical x 2)
Main Title	: Arial Black, 12 pt., Centered
Text (Labels)	: Anonymous, 8 pt., Left
Text (Content)	: Anonymous, 8 pt., Centered

Note that for the sheet label, I denoted it as '1 of 1' since it will only be one sheet of drawing page for this PCB. If multiple sheets or pages are expected, I'll denote it as 'x of x' where 'x' is the total pages, and put the individual page numbers on their own drawing pages.

We're done with the background page.⁷⁹ Now click on the Page-1 tab to go to the main drawing page. Hey! Where's the grid references and title block? Don't panic. The reason is because Page-1 has not been assigned any background page yet. Go to File → Page Setup... and under the Page Properties tab change the Background from [None](#) to [Background-1](#) and presto!

⁷⁹ Don't despair if you find difficulty following the steps to create the background page's grid references and title block. You can obtain these templates from me and work from them instead of creating from scratch. Go to the Preface of this book under [Additional Resources](#) for more details if you have not done so.

DRAWING THE PCB OUTLINE

Drawing the PCB outline is the next logical step, since it provides the defining boundary to house all the component layout symbols which you'll create later on. If your PCB is a simple through-hole type like the one we're doing, only the component side layout will suffice; if it's a surface-mount type with components on both the component and solder sides, then you'll need to draw both!

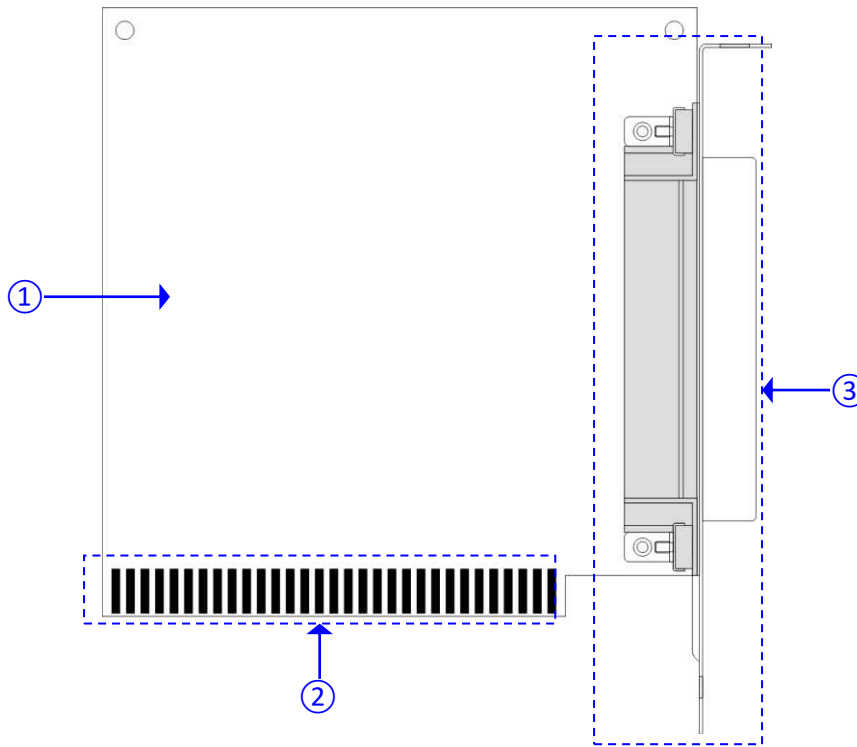


Figure 3.9 PCB outline of the ISA host adapter card.

The half-sized ISA host adapter card is made up of three parts:

1. Main body of the PCB with two guide holes,
2. Gold-plated fingers card edge connector, and
3. PC slot holder with Centronics connector and its base assembly.

At this juncture, let me remind you to save your drawing often and to backup your file, especially if you are using a desktop PC and the power source in your area is unstable and likely to experience frequent outages.⁸⁰ I had an unpleasant experience in which one of my opened work file was corrupted as a result of power shutdown, and it has become inaccessible—whenever I try to open it, Visio would inform me that there is an internal error and come up with a cryptic error number (3400). I've searched Microsoft websites and online forums with people plagued by the same problem, but unfortunately there is no way to recover it. So be warned.

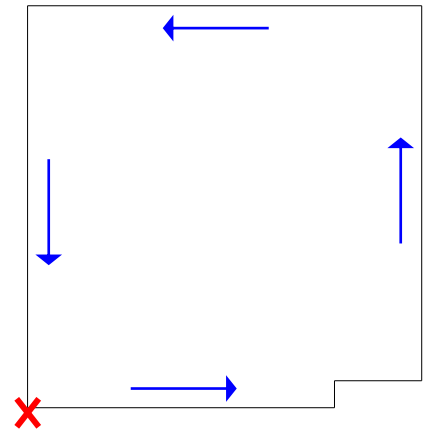
⁸⁰ You may want to consider getting an uninterruptible power supply (UPS) as a preventive measure.

Drawing the PCB Main Body

This portion of the PCB is usually quite straightforward and easy to draw, barring odd-shape boards and those overlaid with a heat sink layer for dissipating heat from ceramic ICs. I'd normally use a ruler with imperial unit markings (tenths) to do estimations of the PCB's dimensions, which is accurate enough for our purpose.

We'll need to decide on the starting point⁸¹ to begin drawing; in our case it will be (2.000,2.500). You can choose either clockwise or counter-clockwise; we'll do the counter boogie. There are altogether six sides to this PCB. The approximate measurements for the six sides are: 3.2, 0.3, 0.9, 3.9, 4.1, and 4.2. You can draw the line as close to the measured length as possible to the nearest grid snap point, but I'll teach you a better and less frustrating way. Open the Size & Position window if you've not already done so. Let's draw it now.

1. Set the zoom level to **200%**. Pan or scroll to 2.000,2.500. Select the **Line** tool. Ensure the Fill property is set to no fill.
2. From the starting point, draw a horizontal line of 0.5 length to the right. With the line still selected, go to the Size & Position window and change the Length to 3.2. Change to the **Pointer** tool (white arrow) and adjust the line's end point to the nearest grid snap point (see note after step 4 below). The length should read 3.1875. De-select line by clicking on an empty area.⁸²
3. From this end point, draw a vertical line of 0.25 upwards. With line still selected, go to the Size & Position window and change the Length to 0.3. Again, adjust the new line's end point to the nearest grid snap point. The length should read 0.2812. De-select line.
4. Repeat this for the next four lines until the last line's end point touches the first line's start point. The four remaining lines should have the following lengths: 0.9063, 3.9063, 4.0938, and 4.1875.



Note: At times it may be necessary for you to zoom in (CTRL + forward mouse wheel scroll) when adjusting the line's end point to the nearest grid snap point. After you're done, zoom back out (CTRL + backward mouse wheel scroll) again to **200%**.⁸³ You should be adept at using the zoom, pan and scroll movements within Visio by now, I hope.

Once you're done, select all the lines, then go to **Shape** → **Operations** → **Join** to join up the lines into an object entity for easier manipulation later on.

⁸¹ This reference point should preferably be the bottom left of the PCB, similar to Visio's origin, so as to facilitate ease of component placement later.

⁸² It is important to de-select the line when you draw the next line, else Visio will join the two lines together and you will not be able to change the length of the new line!

⁸³ This is the default zoom level I find comfortable working in. You should choose one that suits you.

What's left of the PCB outline are the two guide holes:

1. Select the Ellipse tool and draw a circle of size 0.125 x 0.125 and then duplicate it.
2. Select each circle in turn and place them in the following X-Y position using the Size & Position window: 2.1563, 6.5313 (circle 1) and 5.9375, 6.5313 (circle 2).

That completes the PCB outline. On to the next item...

Drawing the Card Edge Connector

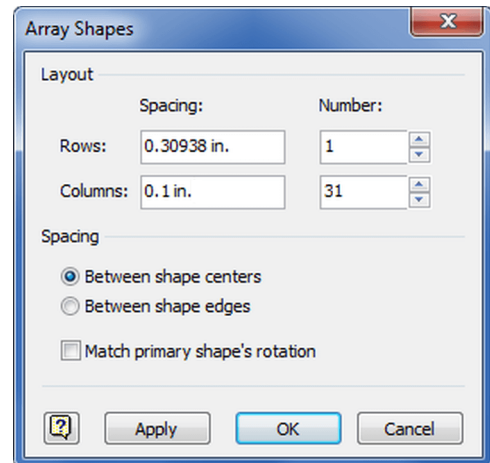
The 8-bit ISA-bus SCSI host adapter has two rows of gold-plated edge fingers,⁸⁴ with 31 fingers on each side labeled A1-A31 (component) and B1-B31 (solder). We only need to draw the component side.



Now, if we were to draw these fingers one by one or even duplicate them, and then align and distribute them, it will take a lot of work. Fortunately, Visio has a special function which I mentioned earlier in this chapter—[Array Shapes](#), which will make light work out of this repetitive task.

Here are the steps:

1. Select the [Rectangle](#) tool and set the fill color to [black](#) (or whichever color you prefer).
2. Draw a rectangle of size 0.05 (W) x 0.25 (H) near to the left bottom corner of the PCB outline where the edge fingers will be located. Change its dimensions in the Size & Position window to 0.06 x 0.3094.
3. With the rectangle still selected, go to [Tools](#) → [Add-Ons](#) → [Visio Extras](#) → [Array Shapes...](#) The dialog box appears as shown. Change the Number for Rows to [1](#) and Column to [31](#). Disregard the Spacing for Rows and change that for Column to [0.1](#). Click OK.
4. Now select all the 31 rectangles and group them into one object using CTRL+G.
5. Finally, with the object still selected, place it at X-Y position [3.5945](#), [2.6734](#) using the Size & Position window.



⁸⁴ Gold finger card edge connectors are still in used today in desktop PC's motherboard, and has evolved from the early 8-bit parallel ISA-bus running at 4.77MHz clock speed to the current serial PCI-Express or PCIe 3.0 standard with multiple lanes blazing data rates at up to 8Gbps per lane.

Drawing the PC Slot Holder

The PC slot holder is a support bracket that holds the card firmly in position when slotted into the motherboard's bus interface connector. Though the bus interface has seen plenty of changes through the years, the PC slot holder has almost remained the same in terms of look and design.

We will draw the support bracket ①, followed by the Centronics connector ②, and then the connector base assembly ③ in that order. Use the Size & Position window to help you replicate the steps, while at the same time try to understand the process you're going through.

Support Bracket

1. Draw the following lines and curves using the data provided (you'll need to zoom in to 1400%):

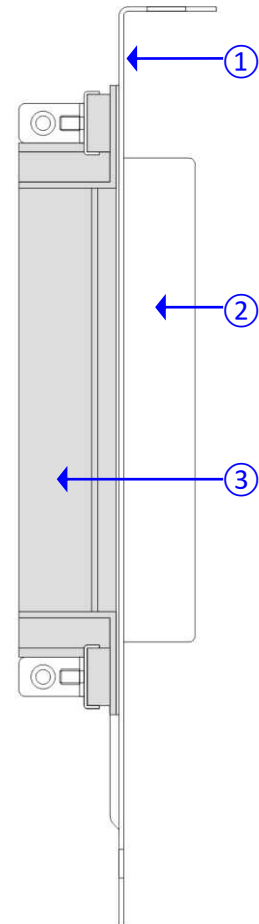
	X	Y	Length	Direction	
Line	6.1563	6.4375	0.4531	Right	
Line	6.6094	6.4375	0.0234	Down	
Line	6.6094	6.4141	0.4531	Left	
	X1	Y1	X2	Y2	Direction
Curve	6.1563	6.4141	6.1328	6.3906	Counter-clockwise
	X	Y	Length	Direction	
Line	6.1328	6.3906	4.7031	Down	
Line	6.1328	1.6875	0.0234	Left	
Line	6.1094	1.6875	4.7031	Up	
	X1	Y1	X2	Y2	Direction
Curve	6.1563	6.4375	6.1094	6.3906	Clockwise

2. Draw two rectangles (no-fill) as follows:

	X	Y	W	H
Rectangle	6.3531	6.4258	0.2000	0.0234
Rectangle	6.1211	2.0117	0.0234	0.1484

3. Draw the side flange as follows:

	X	Y	Length	Direction
Line	6.1094	6.0313	0.0663	Left -135°
Line	6.0625	5.9844	3.7500	Down
Line	6.0625	2.2344	0.0663	Right -45°



Select the lines that made up the flange. Go to **Shape** → **Operations** → **Join**, then **Format** → **Corner Rounding...** and type in 0.0625 for the **Rounding** property. Now zoom out to 100% and select all the shapes that made up the support bracket and press CTRL+G to group them. This completes the support bracket drawing.

Centronics Connector

Drawing the Centronics connector is very much like drawing the flange of the support bracket:

1. Draw the following lines (zoom at 400%):

	X	Y	Length	Direction
Line	6.1328	5.6563	0.3672	Right
Line	6.5000	5.6563	2.5000	Down
Line	6.5000	3.1563	0.3672	Left

2. Select the lines that made up the connector. Go to **Shape** → **Operations** → **Join**, then **Format** → **Corner Rounding...** and type in 0.0625 for the **Rounding** property. This completes the Centronics connector.

Connector Base Assembly

The connector base assembly is a bit more complicated, comprising three sub-parts to create: The spring base plate (1), the body bulk (2), and the two spacer angle plate fasteners (3).

Spring Base Plate

Select the **Rectangle** tool and **white** fill color.

Draw a rectangle of size 0.0156 (W) x 3.25 (H) and position it at 6.1016, 4.4063.

Body Bulk

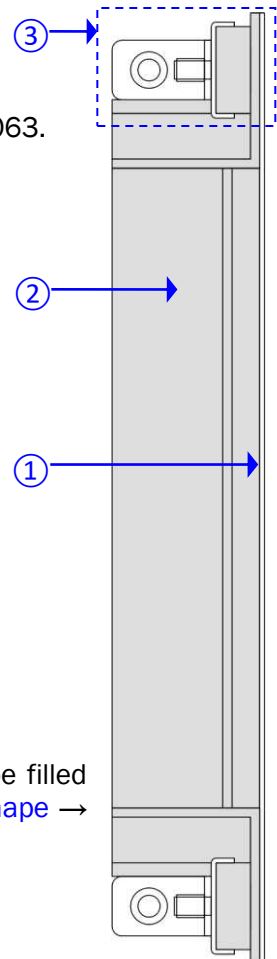
1. Change the fill color to **15% grey shade** and draw two rectangles:

	X	Y	W	H
Rectangle	5.8438	4.4063	0.5000	2.1875
Rectangle	5.9844	4.4063	0.0313	2.1875

2. Select the Line tool and draw the following lines (zoom in at 400%):

	X	Y	Length	Direction
Line	5.5938	3.3125	0.5000	Right
Line	6.0938	3.3125	0.5313	Down
Line	6.0938	2.7812	0.0313	Left
Line	6.0625	2.7812	0.5000	Up
Line	6.0625	3.2813	0.4688	Left
Line	5.5938	3.2813	0.0313	Up

If you draw the lines joined end-to-end, the final reverse 'L' object will be filled with **15% grey shade** automatically. If not, select all the lines and use **Shape** → **Operations** → **Join** to achieve the same effect.



3. Draw the following two rectangles:

	X	Y	W	H
Rectangle	5.8281	3.2031	0.4688	0.1562
Rectangle	5.8281	3.1016	0.4688	0.0469

4. Now group the above two rectangles with the reverse 'L' object in step 2 by selecting them and pressing CTRL+G. Next, make a copy using CTRL+C followed by CTRL+V, and flip the copied object vertically by pressing CTRL+J or clicking the  button. Move the flipped object to X-Y position 5.8438, 5.7656.

Spacer Angle Plate Fasteners

This twin is a bit more work, but fortunately we can draw one and duplicate it for the other as well. So here goes...

1. Change the fill color to [white](#) and draw a rectangle 0.4688 (W) x 0.2031 (H) at position 5.8281, 2.9766. Go to [Shape](#) → [Operations](#) → [Join](#), then [Format](#) → [Corner Rounding...](#) and type in [0.0325](#) for the [Rounding](#) property.
2. Draw two concentric circles of diameters 0.125 and 0.075, center them using [Align Center](#) and [Align Middle](#) if necessary, then group them using CTRL+G. Move them to X-Y position 5.7188, 2.9766.
3. Change the fill color to [15% grey shade](#) and draw a rectangle 0.1172 (W) x 0.2969 (H) at position 6.0039, 2.9766.
4. Change the fill color back to [white](#) again and set the zoom level to [1400%](#).

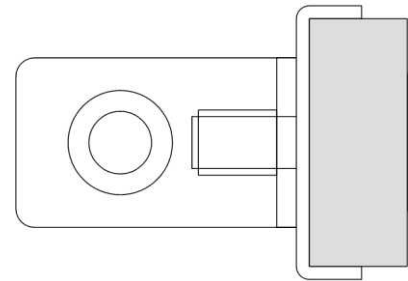
Draw the following lines and curves:

	X	Y	Length	Direction
Line	5.9453	2.8125	0.0625	Right
Line	6.0078	2.8125	0.0156	Up
Line	6.0078	2.8281	0.0625	Left
Line	5.9453	2.8281	0.2969	Up
Line	5.9453	3.1250	0.0625	Right
Line	6.0078	3.1250	0.0156	Up
Line	6.0078	3.1406	0.0625	Left

	X1	Y1	X2	Y2	Direction
Curve	5.9453	3.1406	5.9297	3.1250	Counter-clockwise

	X	Y	Length	Direction
Line	5.9297	3.1250	0.2969	Down

	X1	Y1	X2	Y2	Direction
Curve	5.9297	2.8281	5.9453	2.8125	Counter-clockwise



If you draw the lines joined end-to-end, the final 'C' object will be filled with [white](#) automatically. If not, select all the lines and use [Shape](#) → [Operations](#) → [Join](#) to achieve the same effect.

5. We're left with the screw and restrainer. Draw the following rectangles:

	X	Y	W	H
Rectangle	5.9180	2.9766	0.0234	0.2031
Rectangle	5.8672	2.9766	0.1250	0.0625
Rectangle	5.8594	2.9766	0.0938	0.0781

6. Now select all the objects and shapes created for the spacer angle plate fastener, and press CTRL+G to group them. Then duplicate it using CTRL+C followed by CTRL+V. Zoom back out to 400% for a better view. Move the copied object to X-Y position 5.8281, 5.8359.

To complete the connector base assembly, select all objects and shapes comprising the spring base plate, body bulk, and the spacer angle plate fastener, and press CTRL+G.

Finally, group the three parts (support bracket, Centronics connector, connector base assembly) of the PC slot holder and you're done with the PCB outline. Press CTRL+S to save your work!

Summarizing and Recap

So far, we have created the PCB outline in preparation to lay out the components. I have chosen to give you the X-Y positions and dimensions of the shapes and lines that formed the various parts, which make drawing them so much easier. This is meant to guide you along as you play and familiarize with Visio, while realizing that no matter how complicated an object may be, it can be broken down into smaller and simpler parts that are manageable to draw and re-assemble to its original form.

A few pointers you need to take note are:

1. The faster you become acquainted with Microsoft Visio's environment—workspace and tools—the better you'll be at drawing.
2. You should develop your own style or method of drawing that enables you to stay organized and systematic to become efficient and consistent.
3. Take special care on the order of objects and shapes so that when they overlapped, objects that are supposed to be below stay hidden and not become visible, especially when you do grouping and ungrouping frequently.
4. Often approximation is better than exactness, so learn to leverage on the nearest grid snap to help you draw and position your objects with greater ease. It's a skill which helps you achieve a sense of proportion and balance, without overdoing with time-consuming micro-adjustments or sacrificing accuracy.

Now we are ready to create some component layout symbols!

CREATING COMPONENT LAYOUT SYMBOLS

If you managed to follow the steps in the preceding section without getting lost, or if you did and still managed to recover your composure and complete the tasks, then what you are about to do in this section will be child's play in comparison—in fact, you might even begin to enjoy it already (at least, I hope so!).

Let's refer to the component list for this PCB again:

U1	SN74LS688N	U9	SN74LS688N	C1	100nF	C9	100nF
U2	DIP Switch	U10	74LS244N	C2	100nF	C10	100nF
U3	SN74LS04N	R1	1000	C3	12.6nF, 6V	C11	100nF
U4	DM74LS138N	RP1	007-6718203	C4	100nF	C12	47uF, 6V
U5	Si-P8924	RP2	007-6718203	C5	100nF	C13	47uF, 6V
U6	SP8918	RP3	007-6718203	C6	100nF	J1	Header, 50-way
U7	NCR-5380	F1	1.5A, 125V	C7	100nF		
U8	D2764A	CR1	1N5817	C8	100nF		

You'll notice that apart from the ICs, DIP switch, header and resistor networks, the rest are axial lead discrete components (except C8 which is radial type). We'll begin by drawing these parts.

Discrete Components

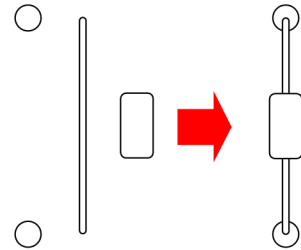
The 100nF decoupling capacitors are the most common, so let's create these first. There is a choice of horizontal or vertical orientation, and since these capacitors are vertically oriented, we shall draw them as they are:

1. Change the fill color to [white](#) and draw a circle 0.06 (W) x 0.06 (H) on an empty area. Use the [Size & Position window](#) to help you if necessary. From this point on, I will not be repeating this reminder so please bear this in mind whenever you need to change the dimensions of shapes or lines.
2. With the circle still selected, go to [Tools](#) → [Add-Ons](#) → [Visio Extras](#) → [Array Shapes...](#) Change the Number for Rows to [2](#) and Column to [1](#). Disregard the Spacing for Column and change that for Rows to [0.5](#). Click OK. Now change to the pointer tool and select both circles, then group them.⁸⁵
3. Next, draw a rectangle of 0.015 (W) x 0.5 (H) beside the grouped circle, and change the corner rounding to 0.0125.
4. Finally, draw another rectangle of 0.075 (W) x 0.15 (H) beside first rectangle, with the same corner rounding of 0.0125. A faster way would be to duplicate the first rectangle and then change its dimensions and save the trouble of doing the corner rounding—a useful tip to remember.

⁸⁵ Again, it bears repeating here that you should know intuitively when to switch to the pointer tool to do shapes selection or manipulation, and to press CTRL+G when grouping shapes without having me to tell you, ad infinitum.

You should end up with three separate shapes lined up as shown to the left of the red arrow.

- Now select all three shapes and perform an **Align Centre** follow by **Align Middle**. Presto! You now have an axial lead capacitor complete with the solder pads. Group the shapes into one object entity. Change the Pin Pos to **Top-Center**.⁸⁶



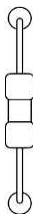
A point to take note when creating component with shapes that overlap one another: you need to take care of the order in which the individual shapes are created and grouped. For instance, if after you created and arrayed the circles but did not group them immediately, but proceeded to create the other two rectangles and then grouped the circles, when you align all three shapes the circles will appear on top of the axial leads instead of below.⁸⁷ If this happens, do not group the three shapes yet—select the grouped circles and press CTRL+SHIFT+B or click  to send it to the back, then select all three shapes to group them. Simple logic!

The next component that's closest to the decoupling capacitor is the resistor. You can go through the same steps outlined above, except that you'll need to make a slight change to the height of the second rectangle, which is the body of the resistor, to 0.25 instead of 0.15. A quicker way would be to duplicate the decoupling capacitor, ungroup it (CTRL+SHIFT+U or click ) then select the rectangular body and change the Height property accordingly, and finally select all three shapes and regroup them again (see footnote 86 below).

The diode or rectifier has straight body edges so besides changing the body dimensions to 0.1 (W) x 0.25 (H), make sure to change the corner rounding to none as well. The leads are also slightly thicker so change it from 0.015 to 0.02 instead. And do not forget to add an additional black bar to the top edge of the body to indicate the cathode end.

The fuse is a tad bit more complicated since the body is capped at both ends by cupped axial leads. You may want to just make do with a simple representation like the resistor, or do the following steps to get a proper fix:

- Duplicate the decoupling capacitor and ungroup the object.
- Select the rectangular body (RB) and change its dimensions to 0.06 (W) x 0.2 (H).
- Duplicate the RB and change its dimensions to 0.075 (W) x 0.07 (H).
- Select the first RB followed by the second RB, then perform an **Align Center** followed by **Align Top**.
- Duplicate the second RB, then repeat step 4 but perform an **Align Bottom** instead.
- Finally, select all shapes and group them (see footnote 86 below).



⁸⁶ The default Pin Pos is **Center-Center**. Set the Pin Pos for every layout symbol you create to **Top-Center**. This will be explained when we come to the section on populating the PCB outline.

⁸⁷ This is also the reason why the shapes need to be filled. Unfilled shapes that overlap will look like tangled wire-mesh instead of creating the correct illusion when a top object/shape overlaps the bottom object/shape.

The radial lead capacitor C8 can be drawn just as easily using the aforementioned method, but without having to draw the leads since they're not visible:

1. Draw two circles 0.06 (W) x 0.06 (H) spaced 0.15 apart using **Array Shapes** and group them.
2. Draw a rectangle 0.05 (W) x 0.15 (H) with corner rounding of 0.0225.
3. Select the two objects, align center and align middle, and then group them.



The last type of capacitors to draw are the axial lead solid electrolyte capacitors with the following geometrical specifications from the datasheet:

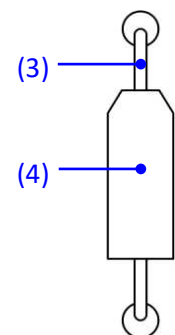
DIMENSIONS in inches [millimeters]			
CASE CODE	D (MAX.)	L (MAX.)	LEAD DIAMETER
U	0.095 [2.41]	0.260 [6.60]	0.020 [0.51]
V	0.110 [2.79]	0.290 [7.37]	0.020 [0.51]
W	0.180 [4.57]	0.345 [8.76]	0.020 [0.51]
X	0.180 [4.57]	0.420 [10.67]	0.020 [0.51]
Y	0.280 [7.11]	0.550 [13.97]	0.025 [0.64]

For this simple PCB, two case code sizes can be found: C3 (V) and C12,C13 (X), and their body diameters and lengths, as well as lead diameter are given. Though the tapered end specifications are not provided, you can estimate them with a ruler with a 30 degree taper angle. So:

C3 (V)

1. Duplicate the decoupling capacitor and place it near X-Y location 3.0, 6.0.
2. Ungroup the object and delete the body.
3. Select the axial lead and change its width to 0.02.
4. Ensure Fill is set to **white** with zoom at **800%**. Draw the following lines:

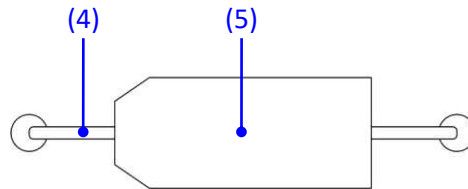
	X	Y	Length	Direction
Line	3.2500	6.2500	0.1094	Right
Line	3.3594	6.2500	0.2500	Up
Line	3.3594	6.5000	0.0456	Up 120.9638°
Line	3.3359	6.5391	0.0625	Left
Line	3.2734	6.5391	0.0456	Down -120.9638°
Line	3.2500	6.5000	0.2500	Down



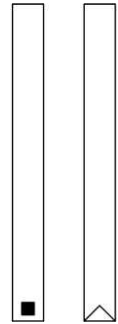
5. The final shape should be filled if you draw the lines joining each other. If not, select all the lines and do a **Shape** → **Operations** → **Join**.
6. Select all objects and do an align center and align middle, and then group them.

C12,C13 (X)

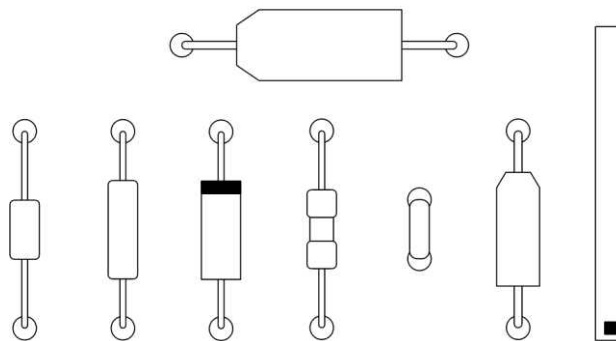
1. Draw two circles 0.06 (W) x 0.06 (H) spaced 0.7 apart using [Array Shapes](#) and group them.
2. Duplicate capacitor C3 and do a rotate left  to orientate it horizontally.
3. Ungroup the object and delete the circles.
4. Select the axial lead and change its dimensions to 0.02 (W) x 0.7 (H).⁸⁸
5. Select the body and change its dimensions to 0.180 (W) x 0.420 (H).
6. Select all objects and do an align center and align middle, and then group them.



The last of the discrete components we want to draw are the resistor networks (RP1-RP3). These are 8-pin SIP containing seven resistors with a common node at pin 1. You can draw them as simple rectangles of 0.075 (W) x 0.8 (H), with a solid square or triangle marking the end where pin 1 is. I'll go with the solid square. There's no need to go too technical on the size of the markings, so long as it looks presentable.



We're done with the discrete components, except the 50-way header (J1) which we'll deal with later on. For now, group these layout symbols, one for each type, in a convenient area on your drawing page, similar to that shown below:



Note that the layout symbol drawings shown are magnified for better viewing and should not be taken as the actual sizes.

Let's move on to draw something more interesting...

⁸⁸ Rotating an object by 90° does not automatically reassign the width (W) and height (H) to its new orientation. Visio still retains the original association, so make sure you understand this when you make changes to the values.

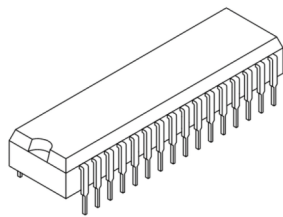
Integrated Circuits

Based on the component list, the following IC packages are present:

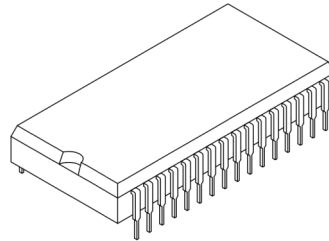
14-DIP 300-mil	U4,U5,U6
16-DIP 300-mil	U3
20-DIP 300-mil	U1,U9,U10
28-DIP 600-mil	U8
40-DIP 600-mil	U7

Through-hole DIP ICs come in two packages regardless of their pin count and material type (ceramic or plastic): the 300-mil or 0.300-inch narrow package, and the 600-mil or 0.600-inch wide package.⁸⁹

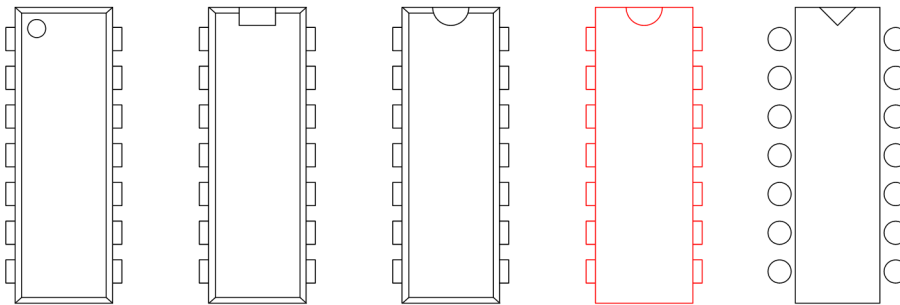
DIP 300-MIL



DIP 600-MIL



Again, there are a number of ways to represent DIP IC package as layout symbols, and some of the more popular ones are shown below:



The first three are more elaborate forms with emphasis on a 3-D look while the last two are simplified forms, the last of which is more of a via-hole artwork⁹⁰ than a physical representation. We will only focus on creating the **fourth** variant for this simple PCB, though you may like to attempt the others on your own once you learn how to do for one of these.

⁸⁹ There's actually a third DIP IC package known as the 400-mil or 0.400-inch package but it is less commonly seen nowadays.

⁹⁰ The via-hole artwork is usually found on bare board PCB, though it can also be used for assembled boards to indicate the positions of the via-holes for all through-hole components. I've seen such PCB layouts and also used them myself in some of my reverse-engineering works (see Figure 3.1).

Below are the physical specifications of a 14-pin dual-inline-package (DIP) which can be found in most IC datasheets:

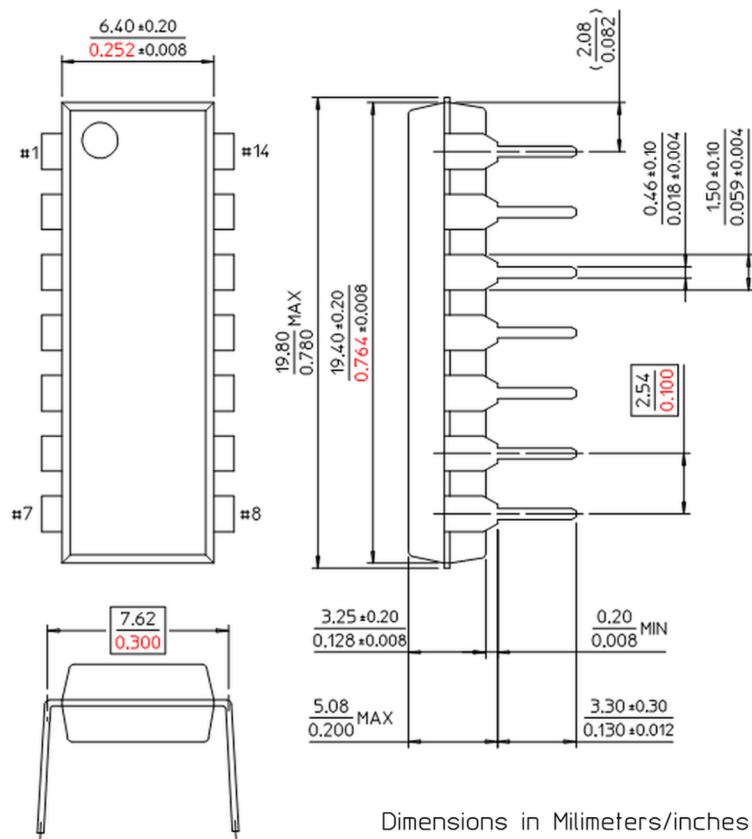


Figure 3.10 Physical specifications of a 14-pin DIP IC.

Notice the dimensions are given in both metric and imperial units. Choose the one you're comfortable working with—I'll stick with the imperial unit (highlighted in red) for our example here.⁹¹ Take note that the legs are shown as slightly flexed ($0-15^\circ$) which is how ICs are fabricated and packaged in long plastic tubing, but when assembled and soldered on PCBs, these IC legs will be bent to conform to the via holes which they are inserted into, at 0.3 or 0.6 inch apart.

⁹¹ Of course, you can make direct measurements with a ruler to draw the IC instead of relying on the datasheets. That's what I did when I started out to do reverse-engineering and then realized that this approximation method works acceptably well with discrete components but becomes difficult on the eyes and brain considering how reliably consistent one can achieve by raw estimation. For a simple PCB with few ICs it may not be evident, but wait till you work on PCBs containing rows and columns of ICs with varied pin counts, and you'll experience the frustration of lining them up accurately like I did. No kidding!

Let's begin by drawing the legs of the 14-pin IC, which are spaced 0.1 inch apart on each side and 0.3 inch across, with a broad width of 0.059 ± 0.004 which we'll take it as 0.06 inch. So:

1. Draw a white rectangle of 0.3 (W) x 0.06 (H).

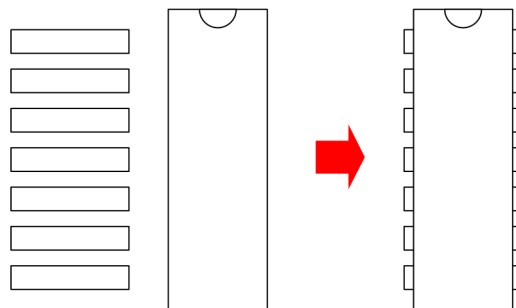
Make sure there is ample room on top of the rectangle before doing step 2. Move the shape to an empty area, if need be.

2. Perform an [Array Shapes](#) with Rows= and Column=1, and a Row Spacing of 0.1. Group the shapes.

3. Draw another white rectangle of 0.252 (W) x 0.764 (H).

4. Draw a semi-circle using the [Arc](#) tool of 0.469 radius at the top edge middle of the second rectangle. Group both shapes.

5. Select the two grouped objects and perform an [Align Center](#) and [Align Middle](#), then group them.



Easy, isn't it? There's no need to create two columns of IC legs since the second rectangle will cover the grouped rectangles to create the illusion of separate leg columns once you align to overlap the objects, taking care to ensure that the second rectangle stayed in front.

The 16-pin DIP IC is a little more tricky to draw—you'll notice that its body size is the same as the 14-pin DIP yet there is an extra leg squeezed in on both sides. Something must make room for the extra pair of legs—in this case, the width of the four corner legs are partially sheared off. There are a number of ways to draw these uneven legs. We'll do the simplest one.

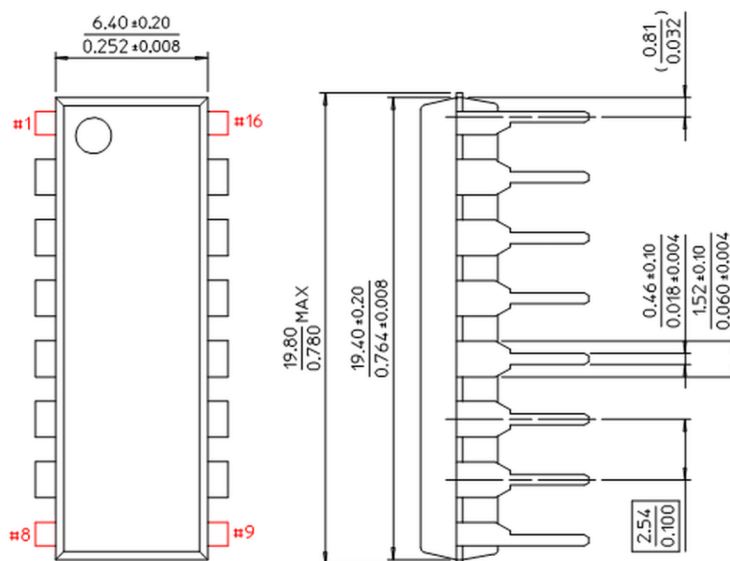
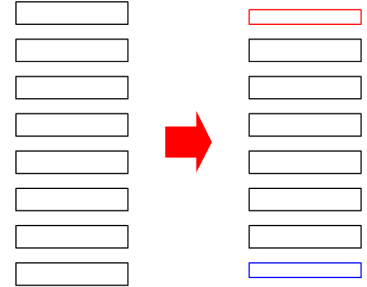


Figure 3.11 Thinner corner pins of a 16-pin DIP IC.

Chapter 3

Based on the specifications given, the corner legs will each have a width of $(0.06 \div 2) + (0.018 \div 2)$ or 0.039 inch. So:

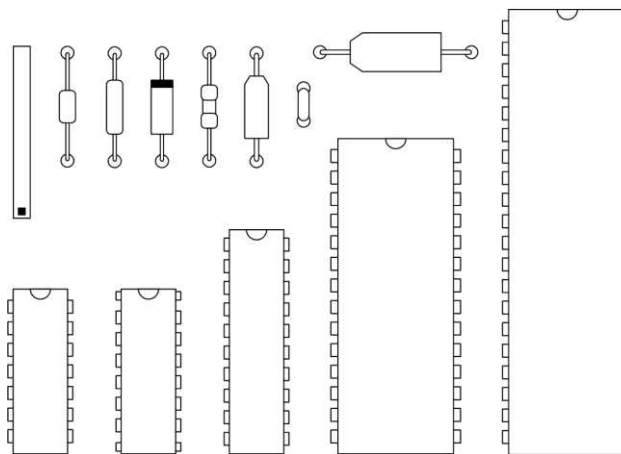
1. Draw a white rectangle of 0.3 (W) x 0.06 (H).
2. Perform an [Array Shapes](#) with Rows=8 and Column=1, and a Row Spacing of 0.1.
3. Select the top rectangle (**red**) and change its Pin Pos to [Top-Center](#). Then change its Height to 0.039 and add 0.009 to its Y value.
4. Next, select the bottom rectangle (**blue**) and change its Pin Pos to [Bottom-Center](#). Then change its Height to 0.039 and subtract 0.009 to its Y value. Group all the eight rectangles.
5. Draw another white rectangle of 0.252 (W) x 0.764 (H).
6. Draw a semi-circle using the [Arc](#) tool of 0.469 radius at the top edge middle of the second rectangle. Group both shapes.
7. Select the two grouped objects and perform an [Align Center](#) and [Align Middle](#), then group them.



Steps 3 and 4 require a bit of explanation. The changing of the Pin Pos for the top and bottom rectangle is to fix one side of the rectangles when changing the height values, instead of having both sides shrunk evenly making it difficult to calculate the up and down Y displacement later. If you understand math, the 0.009 displacement value is self-explanatory.

Using the above steps, it will not be difficult for you to draw the remaining 20, 28 and 40-pin ICs. You'll want to download the IC datasheets to get the physical specifications though.

The total tally of component layout symbols is thus:



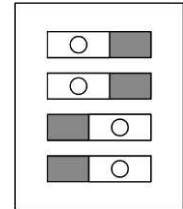
We are left with two components to draw: a 4-way DIP switch and a 50-way header.

Miscellaneous

Most N-way DIP switches come in two mechanical forms: slider and rocker. The rocker switch can either be raised or recessed. The 4-way DIP switch here belongs to the latter type, which requires the use of a pointed tip object such as a pencil or ball-point pen to set or reset the position bits.

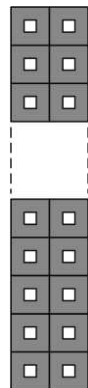
To create the 4-way DIP switch layout symbol:

1. Draw a white rectangle of 0.4 (W) x 0.5 (H).
2. Draw a grey (tint 50%) rectangle of 0.25 (W) x 0.625 (H). Perform an [Array Shapes](#) with Rows=4 and Column=1, and a Row Spacing of 0.1.
3. Draw a white rectangle of 0.15 (W) x 0.625 (H) followed by a white circle of 0.04 (W) x 0.04 (H). Select both shapes, [Align Center](#) and [Align Middle](#), and group them. Then perform an [Array Shapes](#) with Rows=4 and Column=1, and a Row Spacing of 0.1.
4. Select the first grey rectangle and then the first rectangle-circle combo object. [Align left](#) and [Align Middle](#) them. Repeat for the second grey rectangle and the second rectangle-circle combo object.
5. Now repeat step 5 for the third and fourth grey rectangles and their corresponding rectangle-circle combo objects, except this time [Align Right](#) instead.
6. Now select all the four aligned grey rectangles and their corresponding rectangle-circle combo objects and group them into a single entity.
7. Finally, select this entity and the white rectangle in step 1, [Align Center](#) and [Align Middle](#), and then group them.



The 50-way header is made up of two 25 rows of IDC⁹² pins spaced 0.1 inch apart both ways. You only need to create one of the pin and its insulating base, then perform an [Array Shapes](#) operation, as follows:

1. Draw a grey (tint 50%) rectangle of 0.1 (W) x 0.1 (H).
2. Draw a white rectangle of 0.0313 (W) x 0.0313 (H).
3. Select both rectangles, [Align Center](#) and [Align Middle](#), and then group them.
4. Perform an [Array Shapes](#) with Rows=25 and Column=2, and a Row and Column Spacing of 0.1. Group all the 50 objects into a single entity.



This completes the creation of all the layout symbols for the simple PCB's components. Let's proceed to the final task.

⁹² IDC stands for Insulation Displacement Connector, which uses ribbon cables as the means of point-to-point connection. Because the bladed connectors cut through the ribbon wires' insulation when forcefully clamped, thus removing the need to strip the wires of insulation to make connection, it is also known as IPC or Insulation-Piercing Connector.

POPULATING THE PCB

Did I tell you before that the Size & Position window is an important and indispensable tool in Visio? If you can't recall, by now you should have realized its power and usefulness (and even possibly think that you couldn't live without it!).

The good news is, you will make plentiful use of it to populate this PCB. The bad news, if it can be considered bad news at all, is that not all PCBs can be, or should be, populated this way. Firstly, it gets kind of boring and monotonous. Secondly, there are alternative and far more efficient methods to populate other types of PCBs. We will explore some of these methods when we do the more complex example later on. For now, let's focus on this simple PCB.

If you've made it this far, your Visio workspace should look something like this:

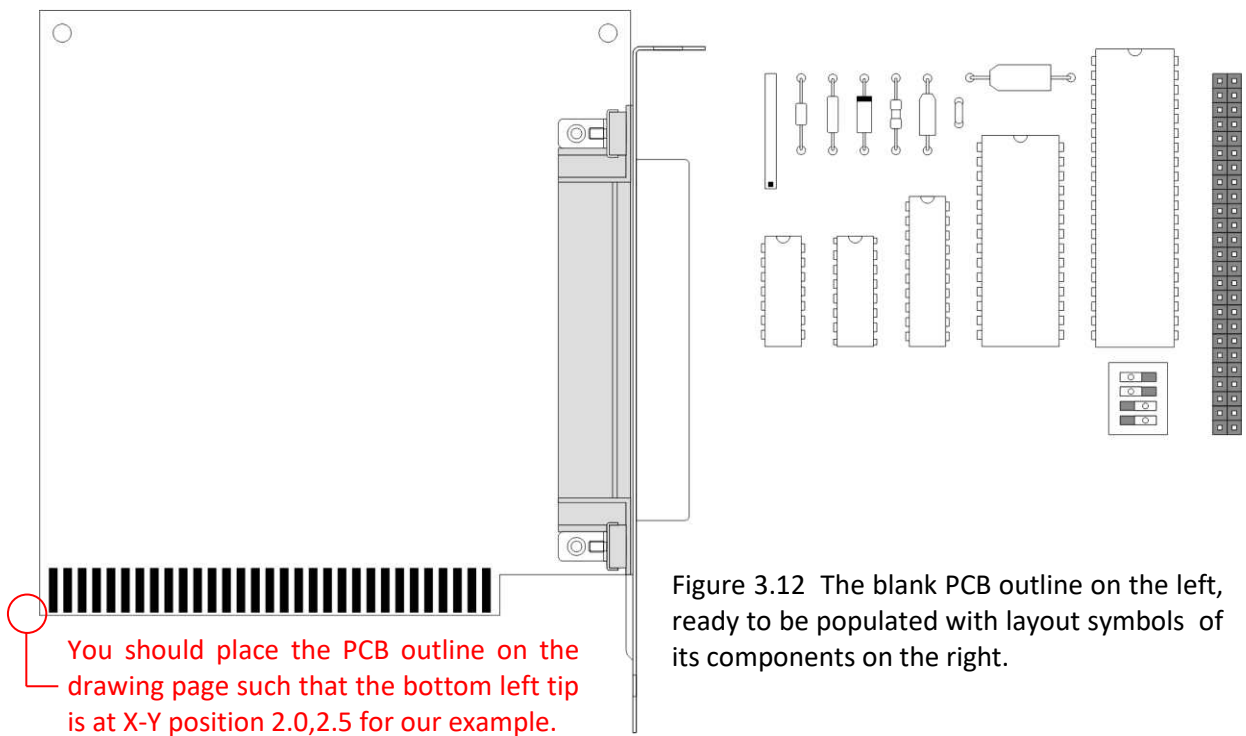
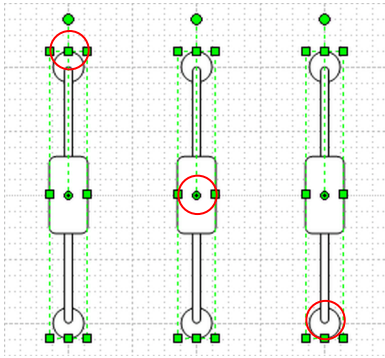


Figure 3.12 The blank PCB outline on the left, ready to be populated with layout symbols of its components on the right.

Before we start placing the layout symbols, here's the Size & Position window again just to refresh your memory. The three properties of the layout symbol we're going to make use of are the X and Y positions,⁹³ and the Pin position which the X and Y take reference from.

X	4 in.	Size & Position
Y	6 in.	
Width	0.5 in.	
Height	0.5 in.	
Angle	0 deg.	
Pin Pos	Center-Center	

⁹³ If you recall, the X and Y position values are with respect to the bottom left of Visio drawing page where the horizontal and vertical rulers start. So X = 4 in. means 4 inches from the left edge of the drawing page, and Y = 6 in. means 6 inches from the bottom of the drawing page.



Refer to **red** circles in figure on the left: To reiterate, the first layout symbol will have its top center handle as the X-Y reference if the Pin Pos is set to **Top-Center**. Likewise, the next symbol will have its mid-center handle as the X-Y reference if the Pin Pos is set to **Center-Center**, and the last symbol will have its bottom center handle as the X-Y reference if the Pin Pos is set to **Bottom-Center**.

You can choose which handle to take reference from but make sure to take your measurements with respect to that handle. I will use the **Top Center** for our example. Let's begin...

1. Set the zoom level to **Page** view.

We shall now place C1 on the PCB outline.

2. Select the decoupling capacitor layout symbol. Check that its Pin Pos is **Top-Center**, if not set it as necessary.⁹⁴ Set the values of $X = (2.1 + 2) = 4.1$ and $Y = (4.0 + 2.5) = 6.5$ ⁹⁵
3. Duplicate C1 and repeat step 2 for decoupling capacitors C2, C4-C7, C9-C11 using the X-Y positions data provided below:

	C2	C4	C5	C6	C7	C9	C10	C11
X	4.725	2.600	2.900	4.100	5.000	2.900	4.100	5.000
Y	6.500	5.600	5.400	5.400	5.400	4.000	4.000	4.000

4. Select the radial lead capacitor (C8). Set $X = 4.1$ and $Y = 4.45$.
5. Select the vertically oriented axial lead solid electrolyte capacitor (C3). Set $X = 4.825$ and $Y = 6.5$.
6. Select the horizontally oriented axial lead solid electrolyte capacitor (C12). Set its Pin Pos to **Center-Left**. Set $X = 2.075$ and $Y = 3.2$.
7. Duplicate C12. Set $X = 4.375$ and $Y = 3.2$.
8. Select the resistor (R1) layout symbol. Set $X = 5.0$ and $Y = 4.7$.
9. Select the diode (CR1) layout symbol. Set $X = 4.95$ and $Y = 6.5$.
10. Select the fuse (F1) layout symbol. Set $X = 5.05$ and $Y = 6.5$.
11. Select the resistor network (RP1) layout symbol. Set its Pin Pos to **Bottom-Center**. Set $X = 5.2$ and $Y = 5.2$. Duplicate RP1 for RP2. Set $X = 5.2$ and $Y = 4.3$. Duplicate RP2 for RP3. Set $X = 5.2$ and $Y = 3.4$.

⁹⁴ **Unless otherwise stated**, you'll need to ensure that every layout symbol's Pin Pos is set to **Top-Center**, else your PCB outline will not be properly populated if the reference is wrong.

⁹⁵ Since you do not have the physical board, I'll be providing the ruler measurements of each component from its top center reference. For C1, horizontal distance from left edge is 2.1 and vertical distance from bottom edge is 4.0, and taking reference from the PCB outline's left bottom X-Y positions of 2.0, 2.5 on the Visio drawing page, a simple addition transformation will place C1's top center reference at **4.1, 6.5**. This working example is meant to show you how I derived at those values; I will only provide the final X-Y data from this point on.

We've completed all the discrete components. Let's do the integrated circuits and miscellaneous parts:

12. Select the 14-DIP IC (U3). Set X = 3.75 and Y = 6.45. Duplicate U3 for U5. Set X = 3.25 and Y = 5.45. Duplicate U5 for U6. Set X = 3.75 and Y = 5.45.
13. Select the 16-DIP IC (U4). Set X = 4.4 and Y = 6.4.
14. Select the 20-DIP IC (U1). Set X = 2.275 and Y = 6.04. Duplicate U1 for U9. Set X = 3.25 and Y = 4.44. Duplicate U9 for U10. Set X = 3.75 and Y = 4.44.
15. Select the 28-DIP IC (U8). Set X = 2.425 and Y = 4.85.
16. Select the 40-DIP IC (U7). Set X = 4.55 and Y = 5.45.
17. Select the 4-way DIP switch (U2). Set X = 2.75 and Y = 6.45.
18. Select the 50-way header (J1). Set X = 5.4 and Y = 6.025.

The fully populated PCB outline should look something like Figure 3.9 overleaf, but without reference designators and text labels. I will not go into detail to provide the X and Y positions of these designators for each and every component. Just one example will suffice:⁹⁶

1. Set the zoom to a comfortable level (400%) and set the font to Anonymous 6pt (if you have not downloaded and installed this font, I suggest you do so now—read footnote 69 on page 61 for the link). Set the text paragraph to align center .
2. Select the Text Tool  and click on an area near the IC layout symbol you want to label (e.g. U1). A text box with blinking cursor will appear. Type U1 and then select the Pointer Tool. Go to Format → Text... and in the Text Block tab set Alignment to Middle, and change the Margins to 0 pt for all. Adjust the text box to a suitable width and height.
3. Drag the text box to the top of the IC layout symbol. To align the text box with the IC, first click on the IC layout symbol, then press shift and click on the text box and perform an Align Center. The align operation always take reference to the shape or object that is selected first. Remember that.
4. Now you can duplicate the text box and change it to U2, U3, etc. and repeat step 3 to label the other component layout symbols. If the text is too long and extends to two lines, just widen the text box sufficiently. If it needs to be rotated anti-clockwise, select the text box and use the Rotate Left button  or press CTRL + L.
5. For the DIP switch labels, you can type DRQ 1, DACK1, DRQ3, DACK3 each follow by an ENTER to create a text box that spans multiple lines, instead of creating four separate text box entities to manipulate individually.
6. For the SCSI PC HOST ADAPTER text, I changed the size to 8 pt.
7. Lastly, the NCR letters are actually three separate rectangle shapes with black fill and white text of Arial 12 pt font. You can use text box with the same effects too.

⁹⁶ I'm sure by now you should know your way in and out of Visio, so I'll leave it to you to fill them out. Come on! Be bold and daring to try and you'll gain a better understanding of the tool and methodology.

Here is the completed PCB layout diagram for your reference:

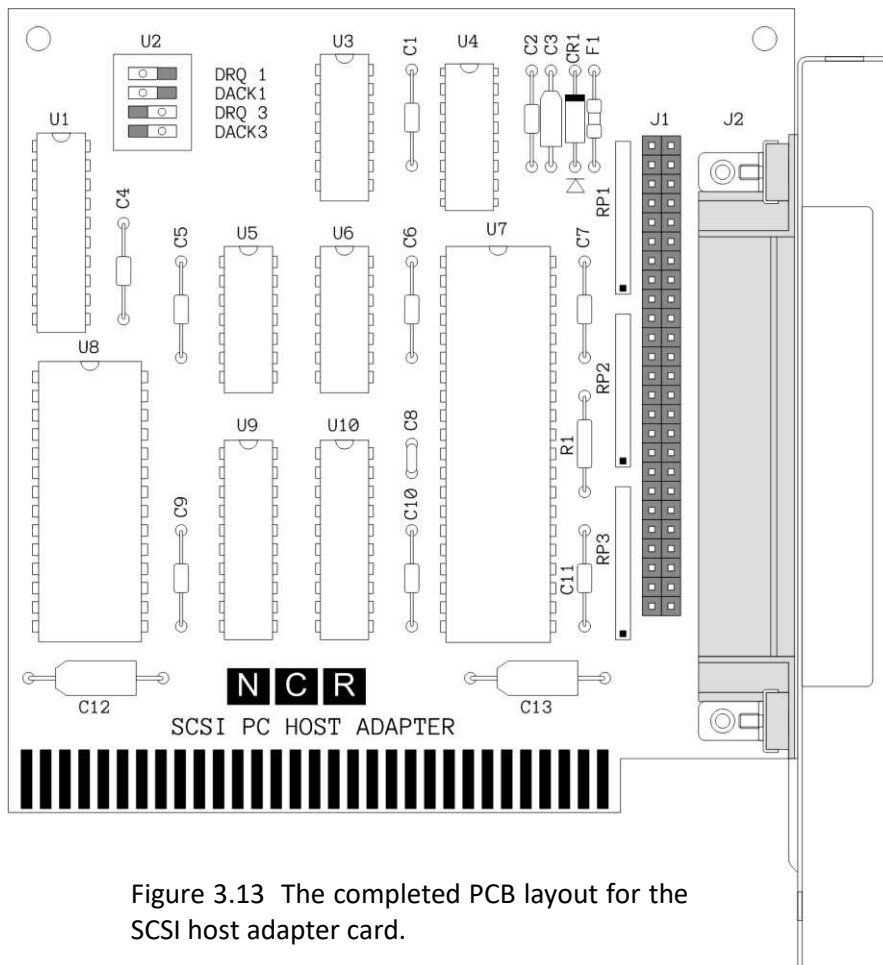


Figure 3.13 The completed PCB layout for the SCSI host adapter card.

It may seem like a lot of work to produce a layout diagram for a simple PCB such as the NCR SCSI Host Adapter card, but in actuality it isn't. In fact, it took me less than two hours to complete the drawing; however, it took me over a week to organize and describe the process as I retraced my steps. Drawing is a breeze, but detailing the steps is hard work, which might explain why many experienced engineers prefer practical hands-on work than writing books.

If you're thinking what we've covered so far is child's play, then be prepared for a bigger challenge. Before you flip the page, take a deep breath and read the following quote:

"A pessimist is one who makes difficulties of his opportunities, and an optimist is one who makes opportunities of his difficulties." Harry Truman (1884-1972)

A COMPLEX EXAMPLE

Now don't get scared and consider bailing out yet. Firstly, I'm not going to put you through the same grilling process as the simple example which we did earlier, for obvious reasons!⁹⁷ Secondly, it doesn't make sense to force you to learn by rote through repetitions of similar instructions. The simple example is meant to get you started and possibly develop your own creative style as you work along, while at the same time to acquaint you with Visio.

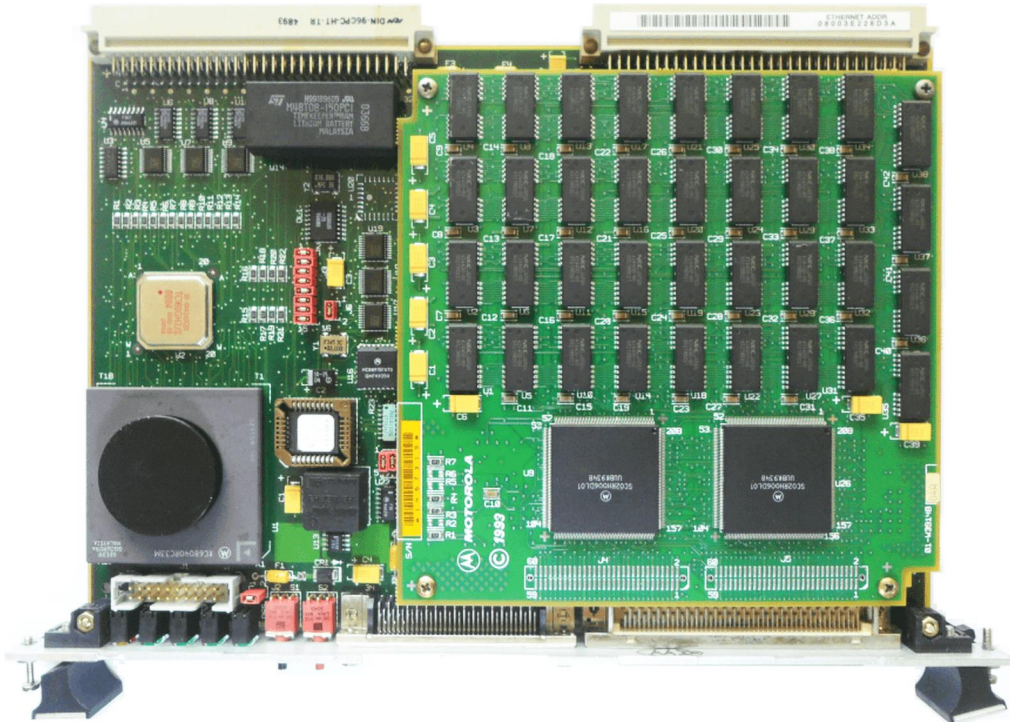


Figure 3.14 The MVME 166-14A Single Board Computer (Motorola)

The main purpose of this section is to introduce you to specific areas of PCB layout that we have not yet covered, using a more complicated board with a greater variety of components. We will learn how to draw complex layout symbols such as the BGA, PLCC, QFP and SOP packages, plus some mechanical parts like toggle switches, connectors, and PCB-mount fixtures, etc., in no particular order.⁹⁸

⁹⁷ If I were to walk you through the same process for this board, it'll probably take up about a hundred pages or more with all the layout symbols' dimensioning and positioning making up at least 75% of the pages—I'm not going to subject you to such an insane and counter-productive feat, even if I have the patience and energy to detail every single step!

⁹⁸ Writing a book of this nature is a tough undertaking, especially for a first time author like me. I'd certainly like to be as comprehensive as I can but in the back of my mind I know there's a limit to how much time and space that can be committed if ever this book is to come to print. The size of this document I'm working on has already exceeded 25MB and we're not even half-way through!

Here's the PCB layout of the MVME 166-14A board (components under the mezzanine memory card have been left out for simplicity's sake):

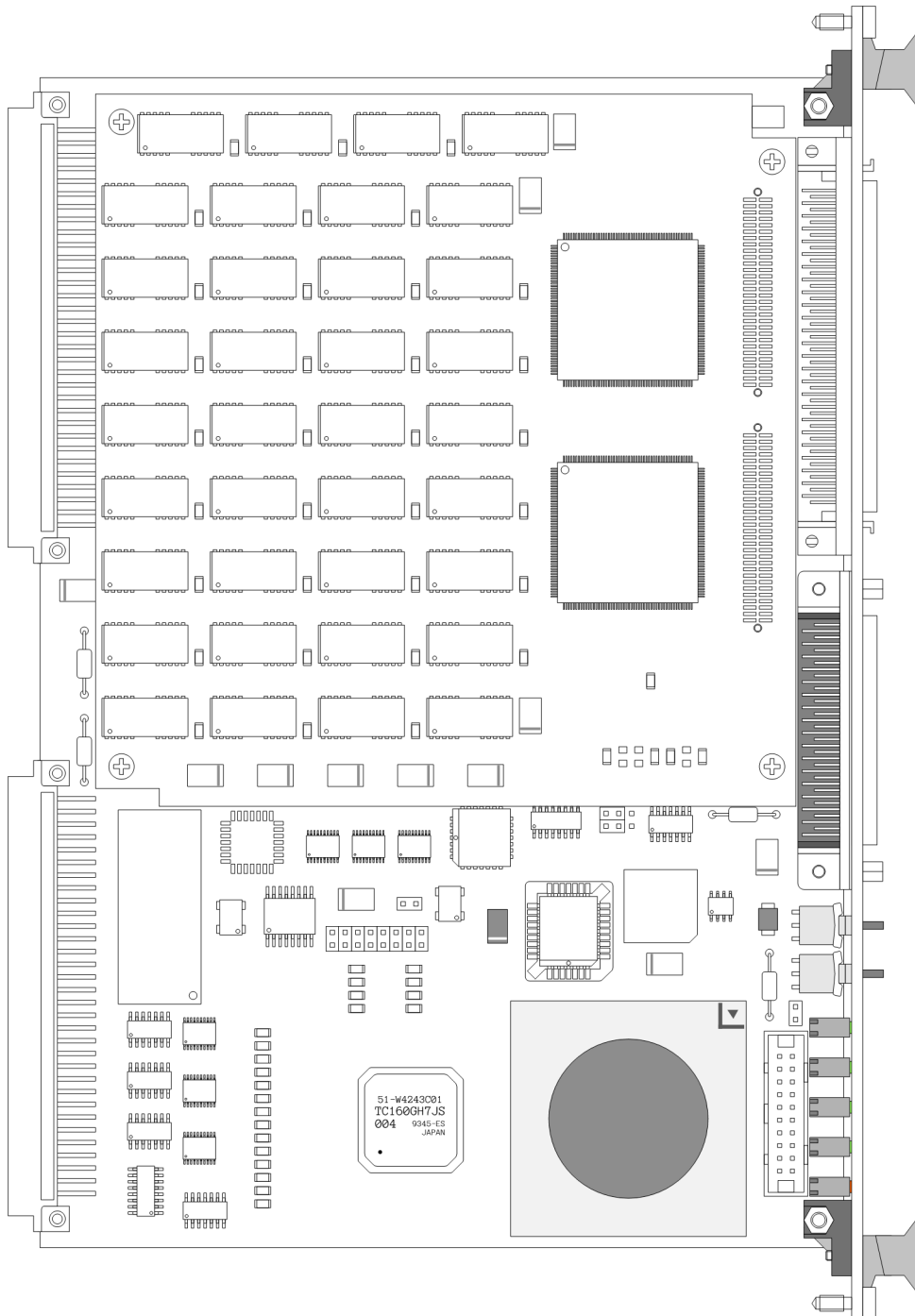


Figure 3.15 Partially completed PCB layout of the MVME 166-14A board.

So how long did it take me to draw the PCB layout? Just slightly over one afternoon, inclusive of tea and toilet breaks. And yes, all the components are drawn from scratch—except for the two switches and part of the card extractor handles. Besides proving that I'm still up to it despite my age and somewhat deteriorated vision, the illustrations will also be used for the topics that I'll be touching on next.

MORE ABOUT CIRCUIT BOARDS

As an electronic engineer, you would no doubt have come across PCBs of various shapes and sizes—standardized or otherwise custom-built. It might still come as a surprise to you that the latter may not be the usual rectangular or straight-edge norm you'd expect a PCB to be, which adds to the difficulty if not interesting challenge of dimensioning and drawing them.

Standardized PCBs, however, are aplenty and common especially in military and test systems. One of such is the VME or Versa-Modular Eurocard board, which comes in four standard sizes:

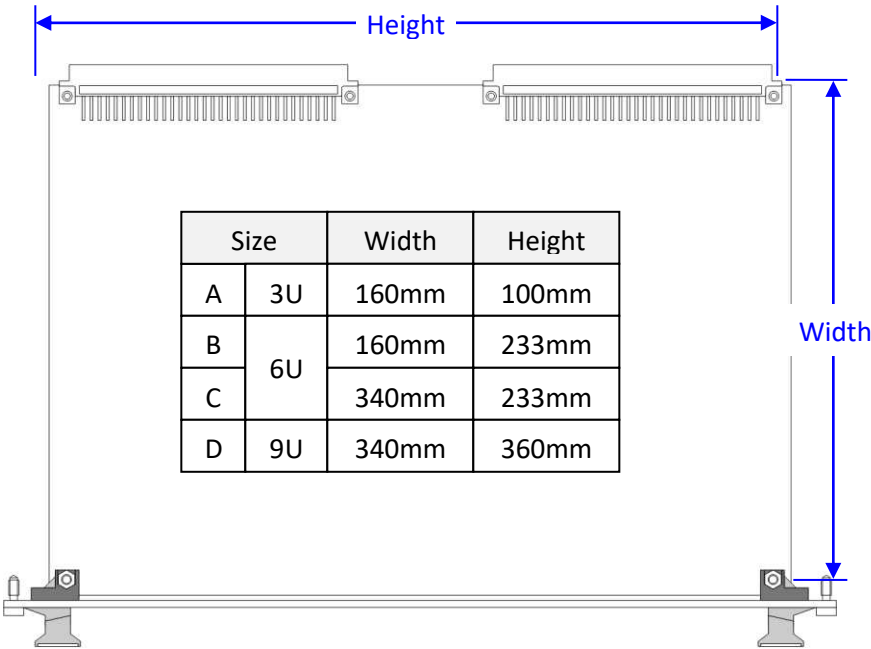


Figure 3.16 VME board sizes and their dimensions.

Notice that there are two references to the size of VME boards, namely A, B, C, D as well as 3U, 6U and 9U. A, B, C and D are basic VXI⁹⁹ terms whereas VME references the board heights by 3U, 6U and the later 9U addition. These boards use what is known as the Eurocard form factors to denote their sizes, which explains why the dimensions are in metric units.

⁹⁹ VXI stands for VME eXtensions for Instrumentation, and as the name implied, is a standard based on the VMEbus for automated test instrumentation. The VME use of 'U' for the board heights is also found in many 19-inch rack and stack systems as well as network servers. 1U is equal to 43.60mm.

I have personally encountered a number of these VME boards containing mezzanine cards (also known as daughter boards) that extend the functionalities of the base board on which they are stacked or affixed to. These are usually commercial boards that are termed COTS or commercially-off-the-shelf cards that provide quick mix-and-match integration and system development. An equivalent stack-based alternative in the embedded computer industry would be the PC/104 standard complying to the 3.550 x 3.775 inches (90 x 96 mm) form factor as defined by the PC/104 Consortium.

Other standard PCB sizes include PCI, PXI and LXI¹⁰⁰ which—like the VXI/VME—are bus architectures and communication protocols which find applications in data acquisition, automated testing, and real-time control systems. Information and specifications on these standards are readily available online, so you should be able to obtain them for your work easily.

PCB-MOUNT FIXTURES

A PCB does not exist alone. Besides the components that populate its physical real estate, it requires mechanical add-ons for the following purposes:

- strengthening overall structure if the board size is big,
- dissipating excess heat if the board wattage is high,
- providing a means to secure the board to the system in which it is plugged into, and
- facilitating ease of removal should the board becomes defective and needs to be replaced.

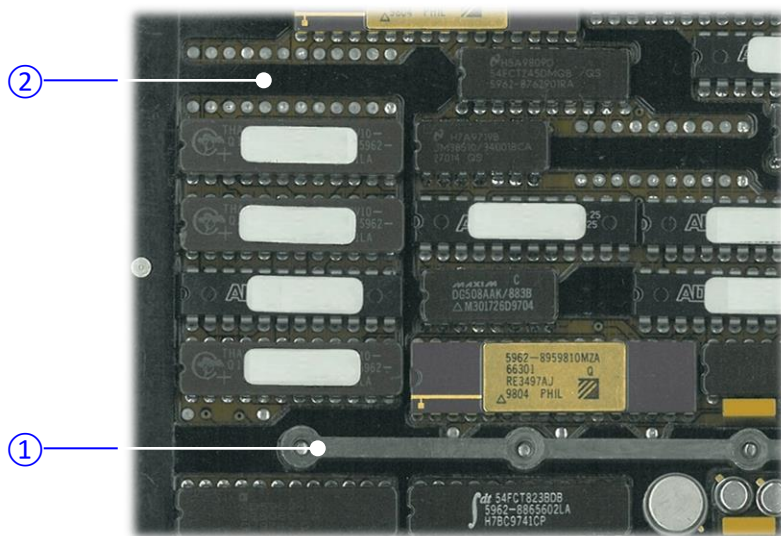


Figure 3.17 Mechanical structure that doubles up as heat sink.

¹⁰⁰ PXI stands for PCI eXtension for Instrumentation and uses the CompactPCI 3U and 6U form factors, and are developed and maintained by the PXI Systems Alliance. LXI stands for LAN eXtension for Instrumentation, and is gaining popularity to VXI and PXI because it offers the size and integration advantages of modular instruments without the cost and form factor constraints of card-cage architectures. (Source: Wikipedia)

As shown in Figure 3.17 above, a metal bar ① can be attached using screws along the center region of the PCB to provide added mechanical support, or a layer of machined plate ② can be permanently affixed by rivets to the top of the PCB that runs along the middle section under each IC to act as heat sink for heat dissipation purpose. In fact, this heat sink layer can perform both functions adequately if the PCB size is small and does not require the use of an extra support bar.

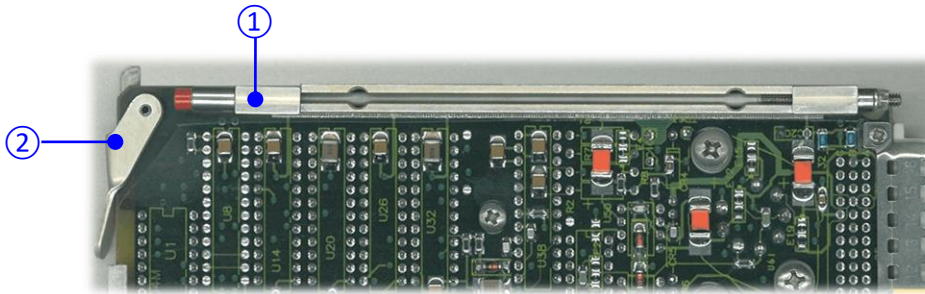


Figure 3.18 PCB wedge-lock retainer (1) and extractor tab (2).

Often, PCBs are either slotted into a card cage by means of card guides lining both sides, or into grooves of a machine-milled casing made from cast metal with external cooling fins for heat dissipation. The former is usually employed in ground-based system where little or no mechanical vibration is present, whereas the latter is found in airborne equipment—in which case a form of wedge lock retainer is used to secure the PCB to the walls of the casing ① by screw tightening with threaded wedges that press against the grooves in which the PCB is seated. To remove the PCB, extractor tabs ② are installed on the top ends of a PCB that act as levers to disengage the PCB connectors from the backplane and eject it from the card cage or casing.

To Draw or Not to Draw?

When dealing with PCBs with these mechanical add-ons, you may wonder if it is necessary to include them in the layout diagram. After all, the PCB layout is only meant to facilitate reverse-engineering activities later on, so from a time-saving point of view just having the component layout symbols should suffice. There's nothing wrong to go on this line of reasoning—in fact, it is valid especially if you do not have the luxury of time on your side.

That said, including these extra elements would certainly make the PCB layout diagram more complete and aesthetically appealing, and give it a touch of professionalism and a further sense of satisfaction to your work. You can opt to include the bare essentials such as the metal bar as a landmark reference and the extractor tabs so the PCB looks more presentable, while giving the heat sink layer and the wedge-lock retainer a miss since they do not serve any practical purpose in the diagram. Or you can simplify their representation instead. It's really a matter of personal style and taste.¹⁰¹

¹⁰¹ I prefer to draw them as they are on the PCB I'm working on, though at times I may leave it to the last or when I have the time to do so—however hard it may be at times. Yup, I'm a perfectionist! Which explains why it's taking me so much time and effort to produce this first book. Good for ya, my readers!

But should you really, really want to accurately depict these parts, here's a sample of how I draw the card extractor for Figure 3.11:

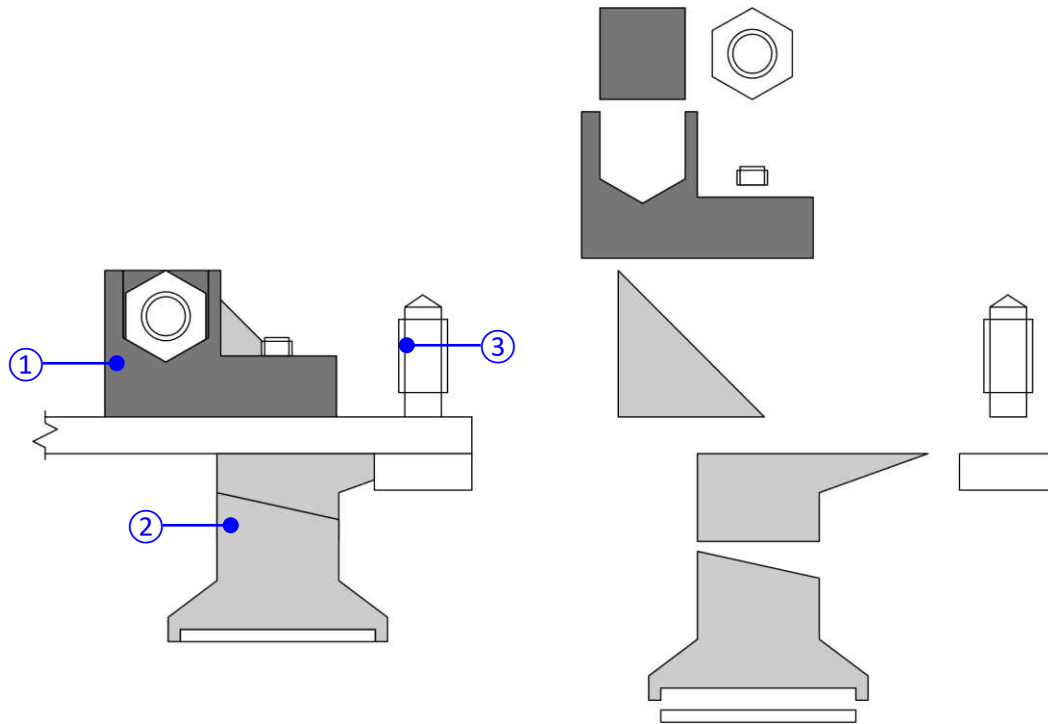
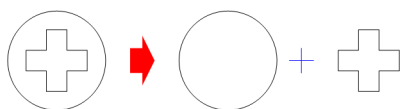


Figure 3.19 Card extractor – assembled and break-ups.

There are basically three parts to the card extractor drawing: ① mount/fastener, ② gripper/extractor, and ③ retaining screw. By breaking them up, it becomes much easier to visualize and draw separately and then re-combine them. I won't go into the details on how to draw them since these are 2-D entities that can be formed from basic shapes or color-filled line-joined objects.

BOLTS AND NUTS

Bolts and nuts are not as difficult to draw as they may seem. Figure 3.15 above depicts two different perspectives—a top view of a bolted nut comprising two concentric circles in the center of a hexagon, readily available from the basic shapes stencil, and a side view which is similarly made up of triangular and rectangular basic shapes. Sometimes a little bit of imagination helps.



As for the bolt's top view, it's simply a basic circle and a line-joined cross combined into one object. You may even rotate the cross 90° or an arbitrary angle to give it that X-factor of irregularity to make your drawing looks authentic!

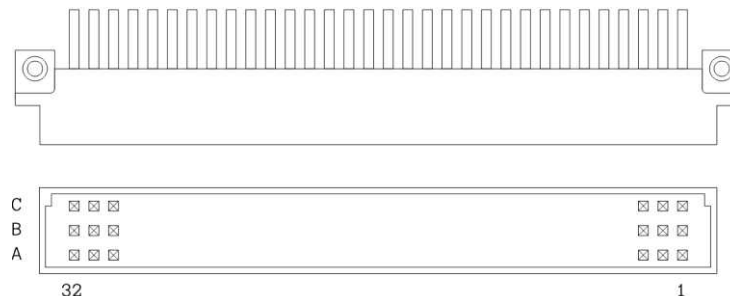
CONNECTORS

Connectors are the means by which the PCB interfaces and interacts with the system it is plugged into. While most connectors are found at the bottom end of the PCB in which it interfaces with the backplane of the card cage, some connectors are located at the top end to allow external interfacing to measuring equipment or peripheral devices. The MVME 166-14A board happens to contain both.

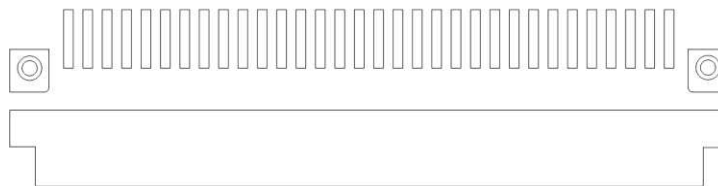
As with most VME boards, the MVME 166-14A uses two standard 96-way DIN connectors for its 6U form factor backend. It has an additional two connectors at the faceplate end: a 68-way SCSI-2 PCB mount connector, and a 100-way I/O interface connector. We will briefly discuss each of these connectors in turn.

96-Way DIN Connector

This is a very common connector that can be found on many PCBs, especially in rack-based systems. The most popular implementations are—you guess it—in VMEbus systems. Though the DIN 41612 standard allows for either 16 or 32 pins per row for up to a maximum of three rows, the 64- and 96-way are the usual with right-angle PCB solder termination pins for added mechanical support, besides the two rivets or screws.



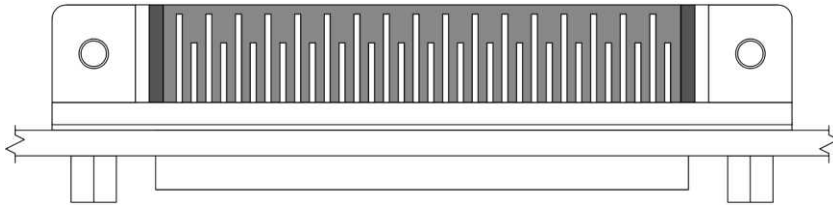
Of course, there is no need to draw the cross-section view of the connector as shown above, which is mainly for illustration purpose. Creating the top view for the PCB layout is not hard either, as can be seen from the break up:



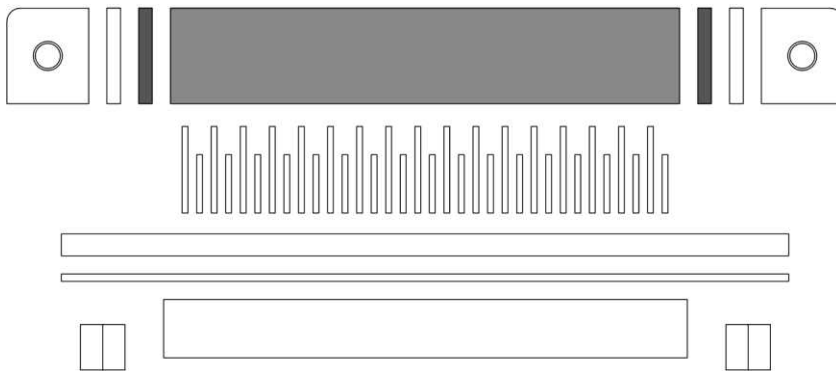
You simply array a row of 32 rectangles of 0.05 (W) x 0.3 (H) each, spaced 0.1 apart, and group them. Then draw the main body and [Align Center](#) the two objects, followed by the two side fastener blocks with white-color fill to overlap the main body. The two concentric circles represent a rivet; you can replace that with a bolt or nut top view if your PCB uses these instead.

68-Way SCSI-2 Connector

The SCSI connector has seen many revisions through the years, from its humble beginning as a low-density 50-way Centronics connector, to the very high-density condensed 68-way Ultra Micro DB68 form. The MVME 166-14A board uses a 68-way SCSI-2 connector, which is a high-density version that lies somewhere in between.



Here's the break up:



I'll not elaborate on the various basic shapes, except for the connector pins which are arrayed in two groups of 17 pins each and of different heights. How do we do that? There are a couple of ways but I'll let you in on how I did mine:

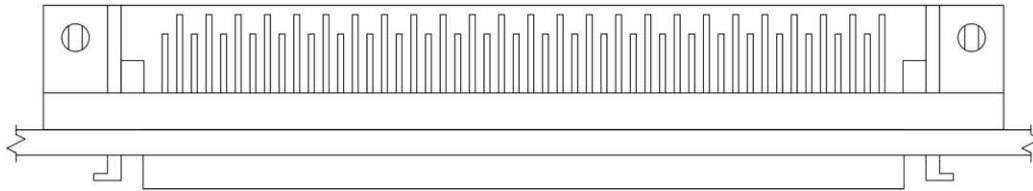
1. Firstly, array a row of 34 rectangles of 0.02 (W) x 0.2969 (H) each, spaced 0.05 apart.
2. Next, press down the [SHIFT](#) key and then click on every alternate rectangle starting from the right—you should end up selecting half of the number of rectangles i.e. 17 of them. Change the height to 0.2. You should see that all the selected rectangles become shortened. Group them.
3. Now select the other 17 original rectangles¹⁰² and group them, so you end up having two separate groups of 17 rectangles. Perform an [Align Bottom](#) and then group the two separate groups into one.

That's it!

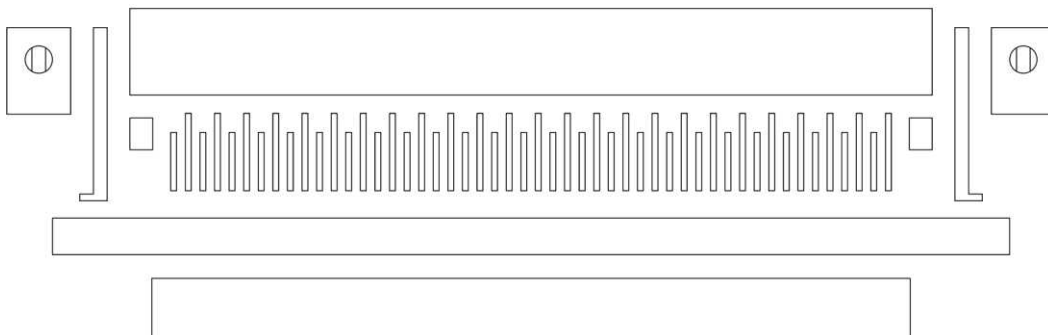
¹⁰² There's no need to individually click on each rectangle like you did the first; just drag a rectangular block around them using the Pointer tool, taking care not to select the first grouped rectangles object.

100-Way I/O Connector

This connector is actually quite similar to the previous SCSI-2 example in terms of the top view:



Again, here's the break up:

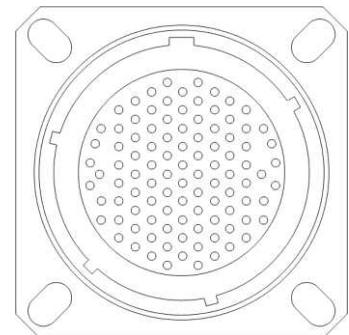


Looks straightforward enough once you break it up into simpler pieces. As long as you take care of the order of the objects, the overlapping will create the right combination of the finished illustration. Need I go through the details? Nah, I doubt so.

[Appendix C-10](#) provides a quick reference of the various types of SCSI connectors available, from the low-density standard 50-way SCSI to the very high-density 68-way SCSI-3, male and female included. Of course, the types of PCB-mount connectors are not limited to the above; in fact, I encourage you to download the product catalogues of manufacturers such as ITT Cannon, Amphenol, Tyco Electronics (TE), Molex, FCI, etc. to get a better idea as well as keep these connector specifications handy for ready reference.

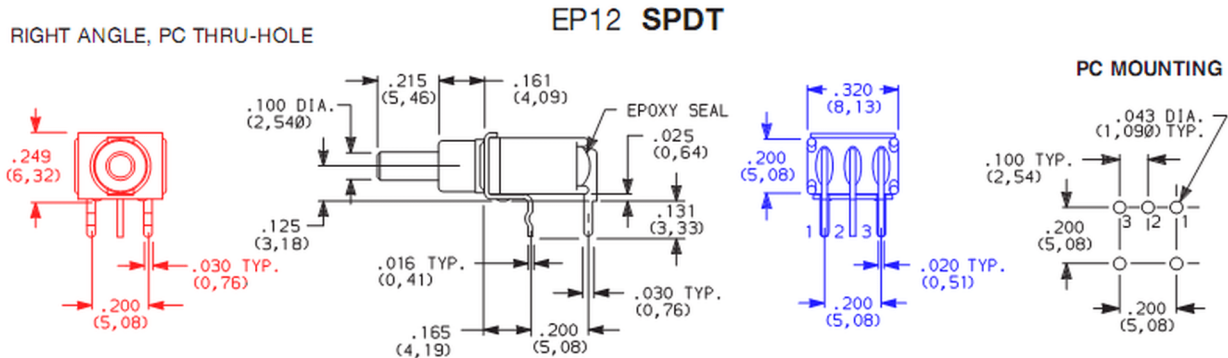
Of particular interest is the circular type connectors, which find typical use in interface cables for military applications. Drawing these connectors do pose a different kind of challenge especially the PCB-mount type used in chassis backplane interfacing.

I've included a sample from one of my layout drawings, a 100-way D38999 series subminiature circular connector complete with flange mounting and keyways. Fortunately, you'll not need to draw these connectors unless you deal with cables and harnesses.

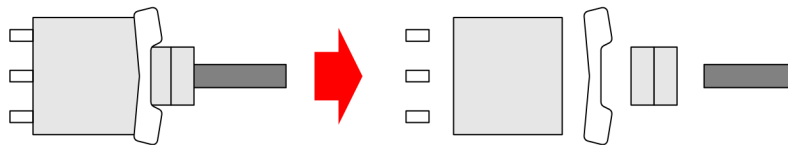


TOGGLE SWITCHES

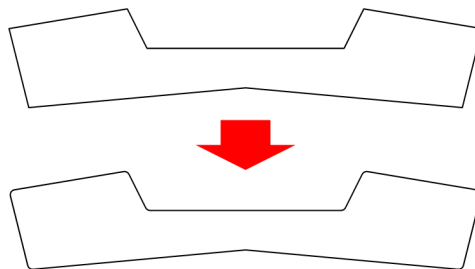
Drawing PCB-mount switches can be quite a handful if you consider the various mechanical details that are tightly coupled in a small unit, as shown in the physical dimension of the sealed tiny pushbutton switch below:



The trick, as aforementioned, is to simplify the parts as much as you can. Depending on the view angle, it may even be possible to omit unnecessary details while retaining the form of the overall presentation. In the case of the MVME 166-14A board, the SPDT switch can be broken down into the following basic shapes from its top view:



Evidently, most of the parts are simple rectangles, except for the switch support bracket which is sort of a bent shape with rounded edges. Again, visual illusion is at work in this instance:



A total of nine straight lines are drawn and joined end to end. You can actually draw one half, duplicate and flip horizontally to produce the other half, then perform a [Shape](#) → [Operations](#) → [Join](#) followed by a [Format](#) → [Corner Rounding...](#) of 0.0125 to finish up the effect. Hopefully, that gives you an idea how to draw other types of switches.

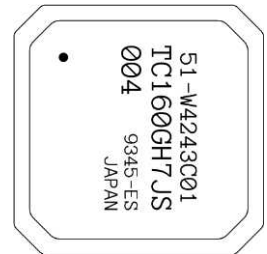
COMPLEX IC LAYOUT SYMBOLS

So far, we have only touch on through-hole dual inline package ICs for the simple example PCB. The MVME 166-14A board has more variety of ICs present, so we'll take a look at each in turn:

Ball-Grid Array

Surprisingly, ball-grid array (BGA) components are quite easy to draw, simply because the solder balls and pads are hidden underneath the IC's body, so effectively you only need to draw the outline of the body and be done with it, unless there are empty BGA footprints on the PCB that are either spares for future expansion or different versions of the PCB.

With increasing use of BGA components on PCBs to fully utilize precious board space and achieve higher speed, reverse-engineering is becoming harder and even to some extent, impossible. The question you'll need to ask yourself when confronted with such a PCB is: do I want to remove that IC to access the solder pads? Re-balling and re-soldering that BGA component will still cost a sum, whether you outsource to a third-party to do it, or if you have the machine to do it yourself. [Appendix C-4](#) provides some sample BGA outlines and footprints dimensioning. Since BGA is an SMT version of the pin-grid array (PGA), I will discuss laying out the footprints in the next section.

**Pin-Grid Array**

Which come first—the chicken or the egg? In the case of the PGA and BGA, there is certainly no dispute. In terms of the top view, the PGA and BGA are no different since through-hole or SMT, the leads and pads are hidden from sight. However, the MVME 166-14A is a mixed type PCB with components found on both sides, so if you intend to reverse-engineer this board, then you'll need to draw the layout for the solder side as well, including the through-hole PGA solder pads.

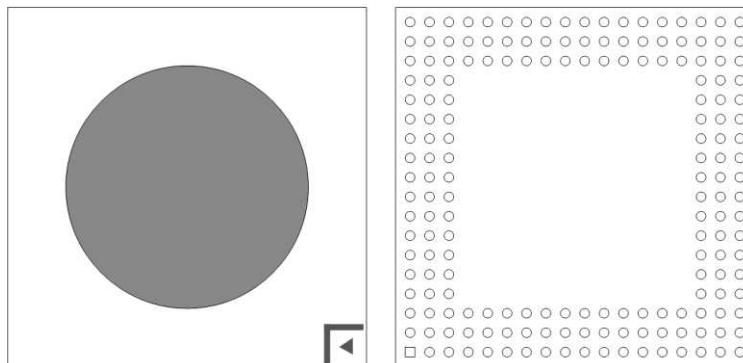


Figure 3-20 180-Pin PGA IC: Top and Bottom views.

Drawing the bottom view of the 180-pin PGA is really a piece of cake—it's just an 18 x 18 array of circles or pads (0.05) spaced 0.1 apart, with the center 12 x 12 array removed:

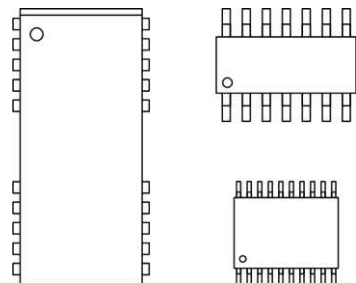


Notice that the bottom left pin is a square instead of circle to denote it as pin A1. Simply draw a square of 0.05 x 0.05 beside the bottom left circle, next select the bottom left circle and bring it to the front (CTRL+SHIFT+F), then press the SHIFT key and select the square, and perform an [Align Middle-Align Center](#) for both shapes. Now click on the overlapped circle and square and press the DELETE key. Since the circle is in front of the square, it will be deleted leaving the square in its place.

Small Outline ICs

Small outline ICs (SOICs) come in two flavors: J-leaded (SOJ) and L-leaded (SOP). [Appendix C-8](#) and [C-9](#) contain sample outlines and dimensioning of these packages.

Inset figure shows a variant of the SOJ IC (left) with spacing between the top and bottom groups of five pins on each side, a standard 14-pin SOP IC (top right), and a shrink SOP or SSOP IC (bottom right) which is a smaller cousin of the SOP IC.



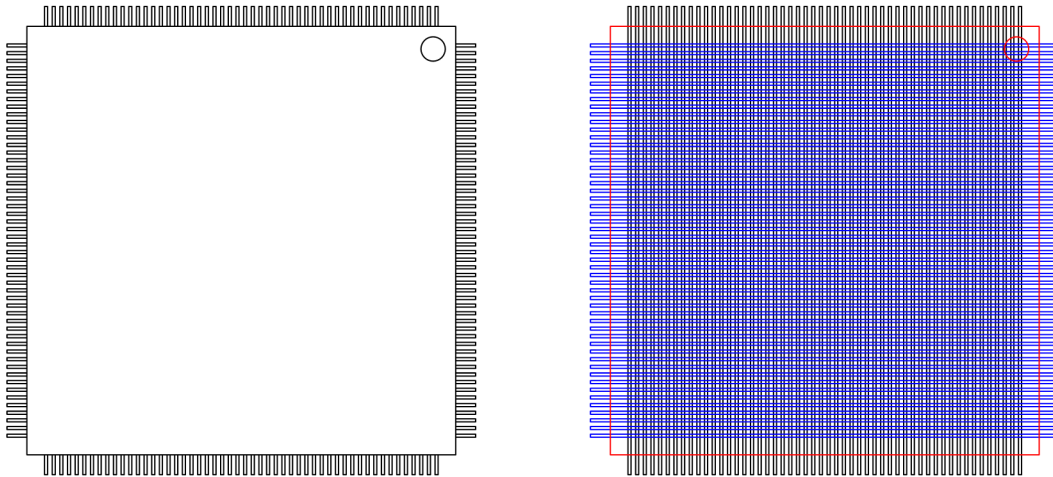
Drawing the SOJ IC is very much similar to the DIP IC, as are the SOP and SSOP ICs with a slight different: you'll need to create an extra shorter set of leads to overlap on top of the longer set.



Special variants of the SSOP include the TSOP (thin small outline package) and TSSOP (thin-shrink SOP) with broad widths and short heights. Most memory components such as the Flash memories are of such packaging, probably due to their die sizes and orientations.

Quad Flat Package

Quad flat package (QFP) ICs are quite straightforward and simple to draw, considering that they have either square or rectangular bodies with equal number of pins on all sides or opposite sides.

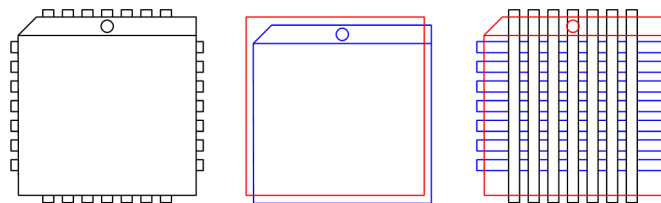


For square type QFP ICs, simply array a row of pins horizontally, group them and duplicate, rotate the duplicate copy, select both groups, perform [an Align Middle-Align Center](#), then draw a filled square body to overlap the pins using [Align Middle-Align Center](#) again, and you're done!

For rectangular type QFP ICs, first array the row of pins for the larger side horizontally, group them and duplicate, rotate the duplicate copy and ungroup it, reduce the pin counts of the duplicate copy to the smaller side and regroup them, then repeat the remaining steps similar to the square QFP, except with a filled rectangular body. It's that simple!

Plastic Leaded Chip Carrier

Of all the IC outlines that I've drawn so far, the plastic leaded chip carrier (PLCC) is the hardest, or should I say more interesting to draw. Like its QFP cousin, PLCC ICs also come in square and rectangular form factors. We'll look at the square PLCC first:



Besides having J-leads similar to the SOJ ICs, the body also has a chamfered side with a cut at the left corner, and a dot at the middle of the chamfered side to denote pin 1 position.

You start by drawing the leads or pins just like the QFP—the body will require a few extra steps though:

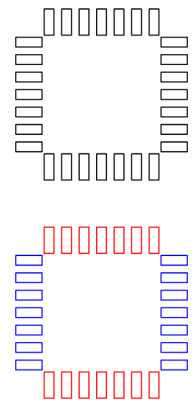
1. First, draw a square (**red**) of the same size to the PLCC body without the corner cut.
2. Next, draw the 45° cut line at the top left corner of the square, then use the **Line** tool to join the rest of the sides using the square as a reference.¹⁰³
3. Draw a line across the top of the new shape from the bottom of the cut line to the opposite straight side, then draw a small circle and **Align Center** it with the line.
4. The new object will overlap on top of the first square but you can still click on the top left corner of the square to select and delete it. Select the new object with the line and circle and group them.
5. Finally, select the body object and the two grouped pin arrays and perform an **Align Middle-Align Center** to finish the layout symbol.

PLCC Footprint

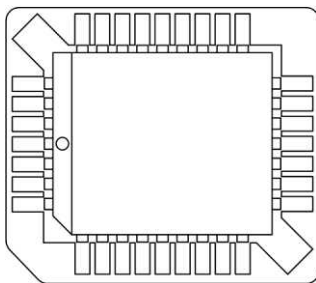
You'd notice in the MVME 166-14A board that there's an empty 28-pin PLCC solder pads footprint. You can choose to draw it or leave it out altogether. Let me assure you that it's not difficult to create, even though Visio does not have the tool to array these solder pads in such a manner.

The trick is to visualize the solder pads as two sets of 2 x 7 array:

1. Create a solder pad of size 0.076 (W) x 0.024 (H) and perform an **Array Shapes** of Rows=7, Columns=2, Row spacing=0.050 and Column spacing=0.449 (**blue** array). Group them.
2. Duplicate the array and perform a 90° rotation (**red** array).
3. Select both arrays and perform an **Align Middle-Align Center**, then group them to complete the footprint.



PLCC Socket

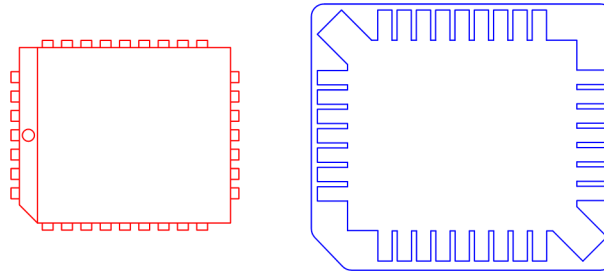


An interesting and challenging object to draw would no doubt be the PLCC socket with a PLCC IC firmly seated in place. If you have been following my line of thoughts and conscientiously worked through the examples given thus far, creating this object should not pose any problem to you by now. I'd bet some of my readers may already have their own ideas on how to break up the object into simpler shapes. However, as a new author I'll be nice and share with the rest how I do it—

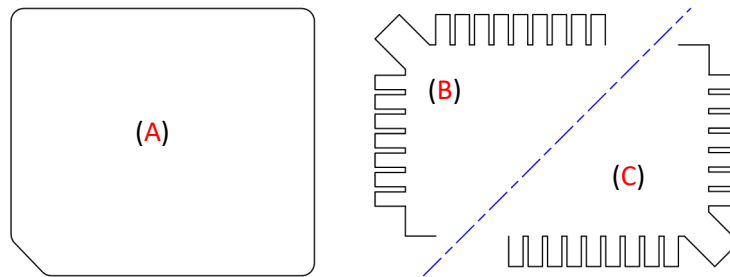
¹⁰³ It's important to enable the snap to shape geometry options in the **Tools** → **Snap & Glue...** menu. Definitely makes your job of joining the lines along the contours of a shape much easier!

Chapter 3

First, you'll need to remove the IC (red) from its socket (blue) and draw them separately:

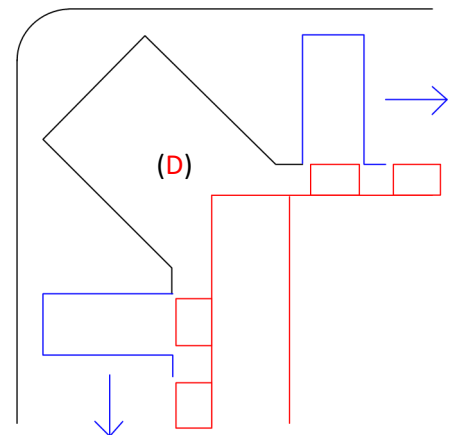


For the PLCC IC, please refer to the previous section on **Plastic Leaded Chip Carrier** for the steps, except that you now replace the square with a rectangle. As for the PLCC socket, it can be further broken up into simpler shapes:¹⁰⁴



Drawing the outline (A) of the socket should be easy if you follow the method I expounded with the PLCC IC body using a rectangle as a reference. Once the lines are joined, use a value of 0.0325 for the corner rounding to achieve the shape mold effect.

To draw the internal structure, it's good to move the PLCC IC body into the center of the socket outline, then zoom in to a maximum level of 1800% so you can draw the lines reliably and comfortably. Start by drawing the corner notch (D), then one horizontal and one vertical slots (blue), and array them in turn so you have nine horizontal and seven vertical slots, respectively. Finally, draw the lower left corner and group all the entities into an object (B). Duplicate it and either do a double 90° rotation or a horizontal flip followed by a vertical flip to achieve an opposite mirror object (C). Join the two objects together to complete the internal structure, and then group them with the socket's outline (A).



This completes the **Complex IC Layout Symbols** section.

¹⁰⁴ I have intentionally limited it to just three though you may want to further simplify it—but I'll try to stick to Einstein's advice (read his quote in the beginning of this chapter to refresh your memory).

DRAWING ANALOG COMPONENTS

So far we have looked at a simple PCB consisting of purely through-hole DIP components, and a more complex PCB containing a richer variety of SMT components. Both PCBs are typical of digital boards with standard IC form factors and manageable discrete components make up. There are times when you may have to work on a PCB that is purely analog or a mixed PCB with a larger number of analog components, like the one shown below:

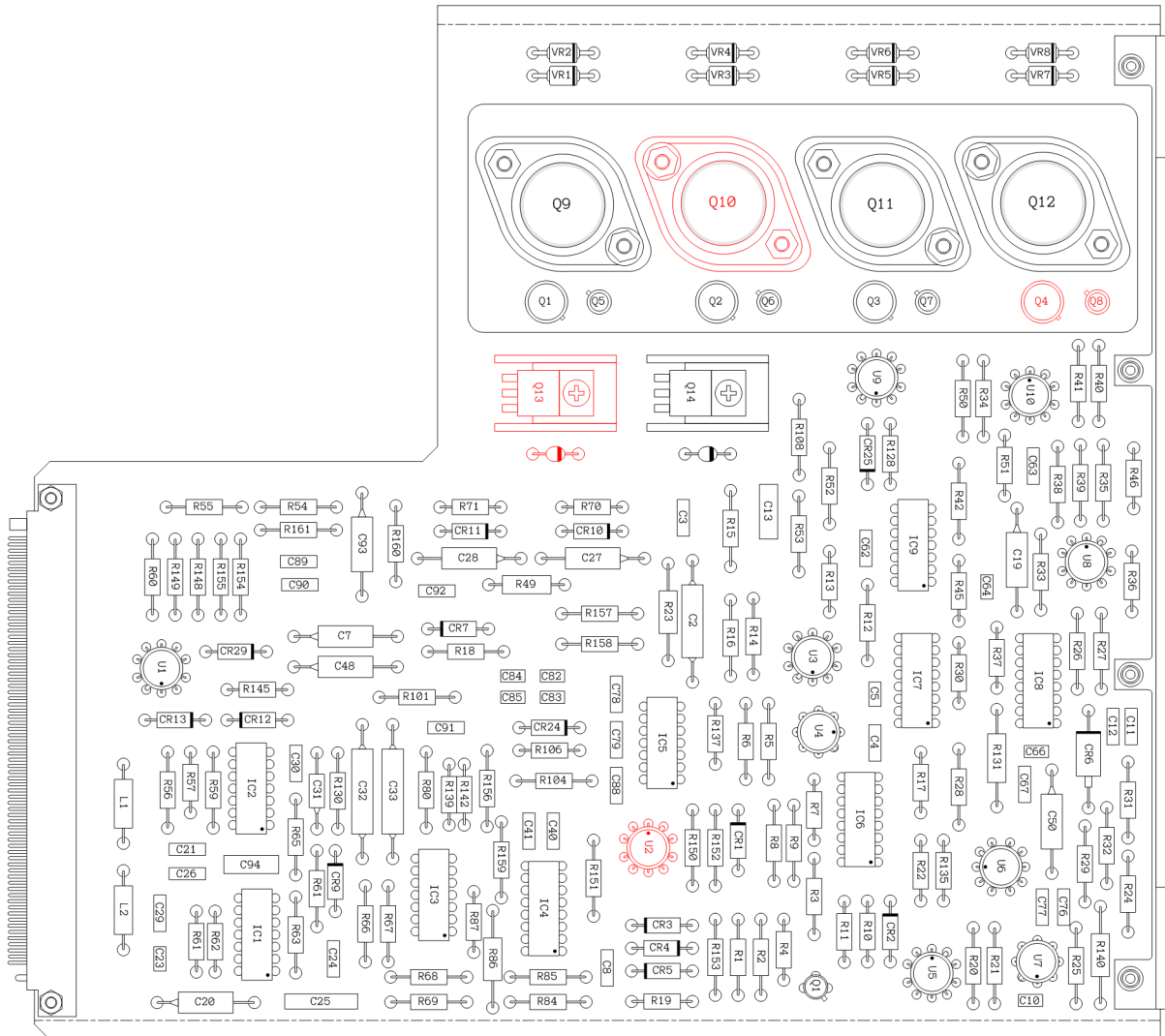


Figure 3.21 An analog PCB with ICs and discrete components.

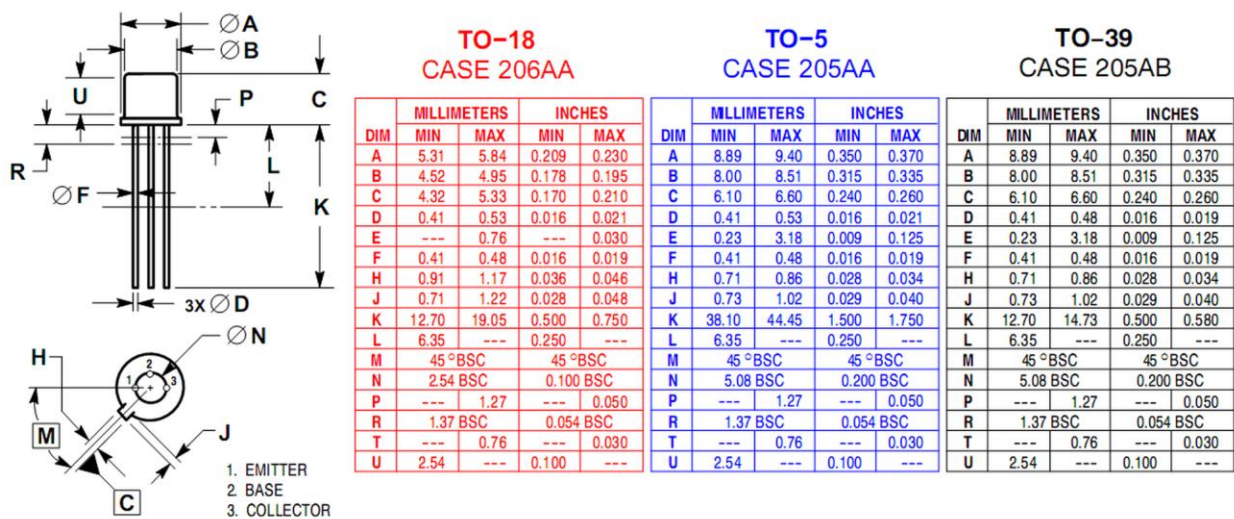
While we have tried our hands on simple discrete components such as resistors and capacitors, there are some analog components that are not that straightforward to draw, such as the TO3 package and metal can analog IC. We will look at some of these devices (shown in red) in the following sections.

Types of Semiconductor Packages

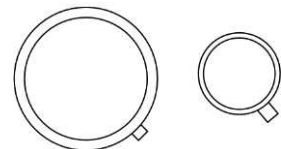
Appendix C-11 provides the packages and pinouts of common transistors (and MOSFETs) with sample part numbers for both NPN and PNP types of these packages. Of interest are the TO-5, TO-18, TO-220 and TO-3 which you'll most likely encounter.

Transistors and Heatsinks

The TO-18 and TO-5 are standard transistor metal can packages, the former being the smaller version which is meant for small-signal switching, while the latter is of a bigger¹⁰⁵ build suited for higher current application. Some transistors come in both packages, such as the 2N2222A (NPN) and 2N2907A (PNP). The outlines of the TO-18 and TO-5 (TO-39) are contrasted below:



From the physical specifications, we can see that the TO-5 and TO-39 are quite similar in sizes with only a slight difference in the leads dimensions. These metal can transistors are quite easily represented using two white-filled concentric circles and a rectangle with a 45° rotation.



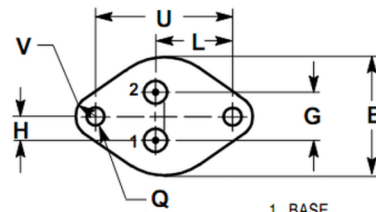
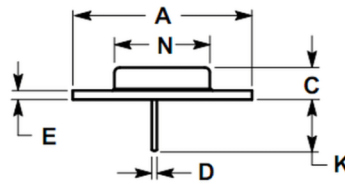
Sometimes these transistors may have mini heatsinks conforming to their body diameters strapped to them as a precautionary measure to prevent thermal runaway when operating in their linear ranges. These mini heatsinks can be of intricate design patterns so you may choose to leave them out and just place a note beside the transistor to indicate their presence.

¹⁰⁵ The TO-5 transistor is also known as a semi-power transistor, whereas the TO-3 transistor is termed the power transistor for its larger body size and thicker leads. Depending on the power and current ratings, some form of heatsink may be necessary for proper heat dissipation to prevent damage due to over-heating. Some TO-3 transistors are even mounted directly on chassis instead of heatsinks due to space constraints.

The TO-3 package is what I would like to emphasize next.

Drawing this symbol requires a bit more thought; and having some background in technical drawing would certainly be an advantage.

I will outline the steps using illustrations follow by narration since this approach is much clearer. Note that dimensions used are based on the average of min and max values stated in the datasheet.



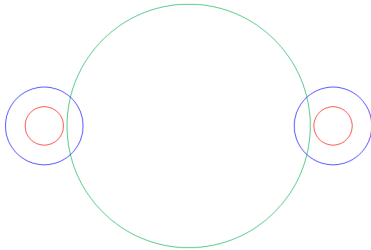
1. BASE
2. EMITTER
CASE: COLLECTOR

**TO-204 (TO-3)
CASE 1-07**

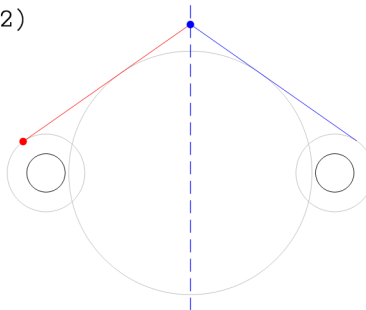
DIM	INCHES		MILLIMETERS	
	MIN	MAX	MIN	MAX
A	1.550 REF		39.37 REF	
B	---	1.050	---	26.67
C	0.250	0.335	6.35	8.51
D	0.038	0.043	0.97	1.09
E	0.055	0.070	1.40	1.77
G	0.430 BSC		10.92 BSC	
H	0.215 BSC		5.46 BSC	
K	0.440	0.480	11.18	12.19
L	0.665 BSC		16.89 BSC	
N	---	0.830	---	21.08
Q	0.151	0.165	3.84	4.19
U	1.187 BSC		30.15 BSC	
V	0.131	0.188	3.33	4.77

Here are the step illustrations:

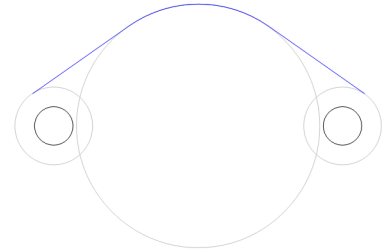
(1)



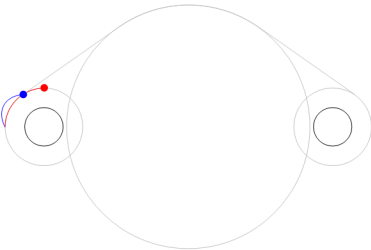
(2)



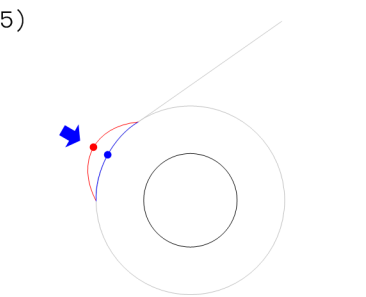
(3)



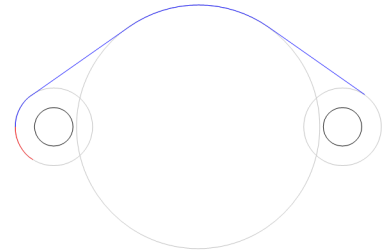
(4)



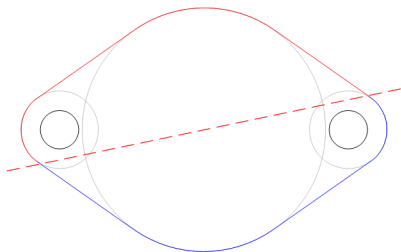
(5)



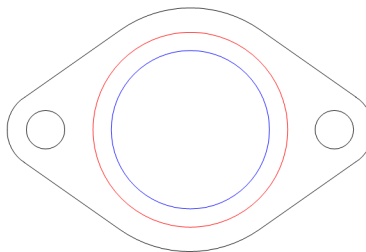
(6)



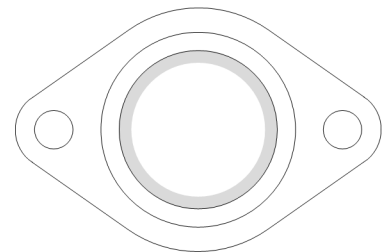
(7)



(8)



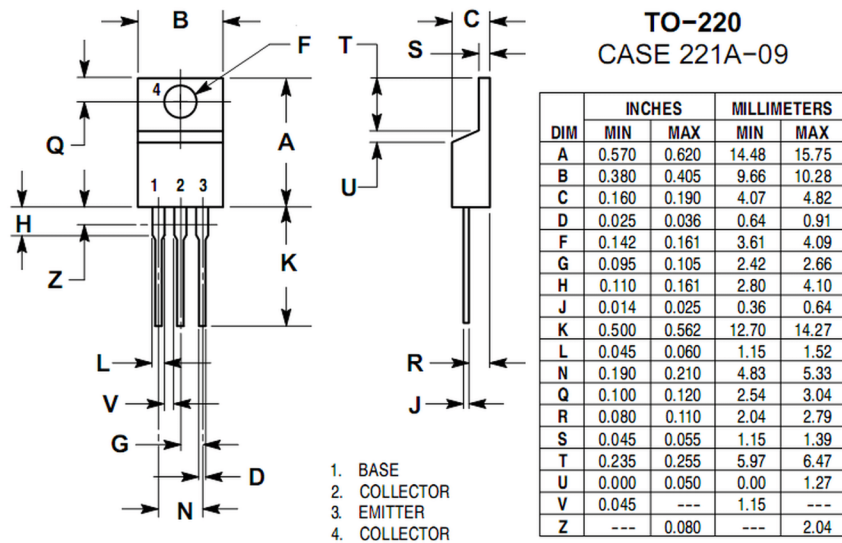
(9)



1. Draw a circle (**red**) of 0.158 diameter and **Array Shapes** at Row=1, Column=2, Column spacing=1.187. This will be the permanent securing holes for the TO-3 package. Next, draw a circle (**green**) of 1-inch diameter and space it between the two smaller circles. This will be the guide for the TO-3 body. Draw another two circles (**blue**) of 0.319 diameter each and align them so they form concentric circles with the two smaller circles. These will be the guide for the TO-3 outer skirting.
2. Select the **Line** tool and move the '+' cursor to the top of the left outer circle until the prompt **Snap to Geometry** appears. Click and drag a line towards the top of the middle big circle where the center (**blue dotted line**) is. Now choose the **Pointer** tool and move the line up or down at the other end until its body touches the circumference of the big circle. In mathematical terms, the line is tangent to the small and big circles now. Duplicate the line and do a horizontal flip, then move the new line (**blue**) so one end touches the vertex of the first line (the prompt **Snap to Vertex** will appear while you hover the new line such that one of its ends touches the vertex end of the other line).
3. Select both lines and perform a **Shapes** → **Operations** → **Join**, followed by a **Format** → **Corner Rounding...** with a value of 0.5. The pointed joint of the lines should now conform to the contour of the big circle.
4. Select the **Arc** tool and draw a quarter arc (**red**) that conforms to the left outer circle's contour. Change back to the **Pointer** tool and drag the top end of the arc towards the vertex end of the joined lines. Release the drag when the **Snap to Vertex** prompt appears. The arc (**blue**) should become elongated as a result of this action.
5. With the elongated arc (**red**) still selected, drag its middle handle towards the outer circle until the arc (**blue**) again conforms to the outer circle's contour. You can fine tune for accuracy by using the **Size & Position** tool to change the Height property of the arc (I used a value of 0.02).
6. Duplicate the arc and do a vertical flip, then move the new arc (**red**) to snap to the vertex of the first arc. Now select the joined lines and arcs and group them.
7. Duplicate the grouped object (**red**) and perform two rotations or a horizontal flip followed by a vertical flip. Move the new object (**blue**) to join up with the other object. Use the keyboard's arrow keys to fine tune its position. There may be a slight overlap but that's OK. Now group the two objects together with the securing holes and then delete the three guide circles.
8. Draw two circles of diameters 0.8 (**red**) and 0.65 (**blue**), respectively, and perform an **Align Middle-Align Center** with the interim TO-3 outline.
9. If you want a 3-D look and feel, color fill the 0.65 diameter inner circle with a shade of grey, then create a white-filled circle without line and place it inside the grey-filled circle.

¹⁰⁶ Zoom to a comfortable level whenever necessary for easier drawing and finer manipulation of the shapes.

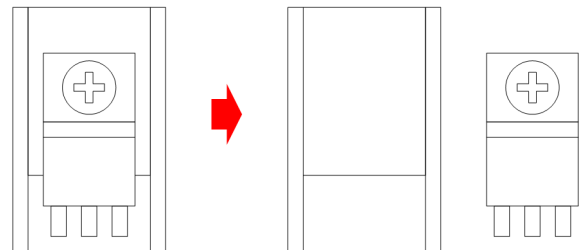
One last transistor package to consider is the TO-220:



Most often, packages of this nature are used for 3-pin devices such as the 78xx and 79xx series voltage regulators, and power transistors or MOSFETs found in switching power supplies. As such, they are usually accompanied by heatsinks on which they are affixed to, or directly mounted onto the chassis of power supply modules.

Fortunately, it's not difficult to draw these packages as they can be constructed using basic shapes and lines including the heatsinks.

Note that the leads or legs are usually bent 90° if the device is lying down on the PCB so there's no need to draw the thinner parts.



Diodes

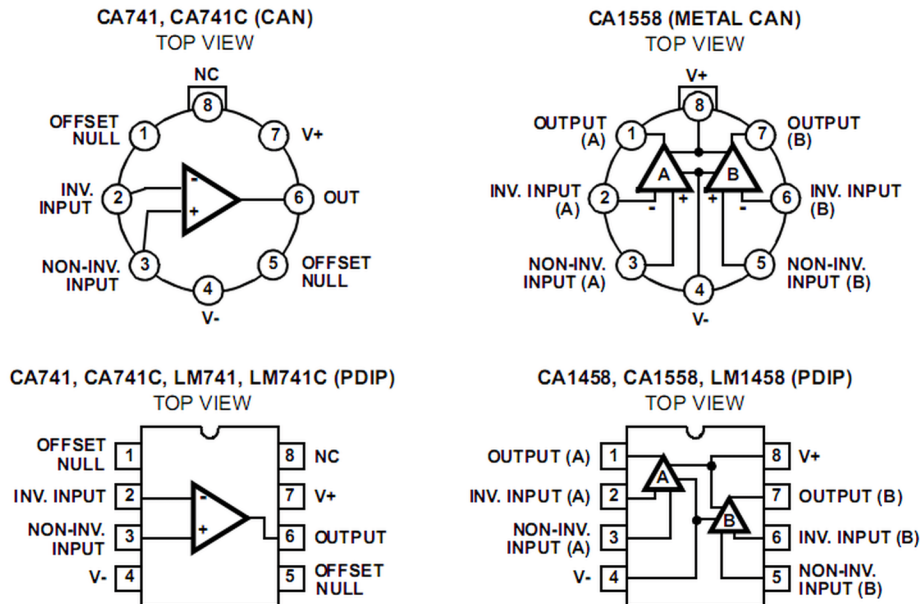
It may seem strange to revisit this common part here, but it just came to mind that like the transistors, diodes or rectifiers do have their peculiarity in terms of packaging as well.

One of these is the glass passivated type which looks very much like a bead strung up in a lead wire. Not that it's hard to draw, but the name is worth mentioning since I recalled having a tough time locating its specifications because I did not know what kind of a diode it was back then.



Metal Can Analog ICs

It is good to know that ICs don't just come in the standard DIP or SOIC packages that we talked about earlier on. Analog ICs such as operational amplifiers do come in metal can as well, since they are not as densely fabricated due to their analog functionality requirements. Examples are the CA741 (single) and CA1558 (dual) opamp ICs:



In such instances you are faced with the prospect of drawing IC pads that are arrayed in a circular manner, and unfortunately Visio does not have a tool that will allow you to do just that, unlike the row-column arrays we've been doing so far. There are people asking in online forums regarding this and the solution suggested is some kind of VBA programming snippet which copies a selected shape and place it in a polar (circular) array.¹⁰⁷

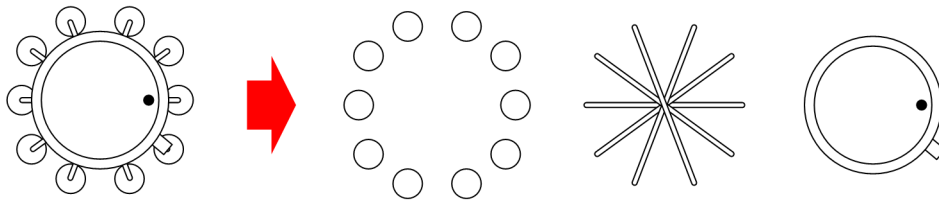
A Visio MVP by the name Visio Guy posted a rather innovative way of working around this using what is known as the radial elements tool, but like he said in his own words, "Visio has the tools to draw circular-arrays of objects, but learning to do it takes time. There are no direct 'arrange stuff in circles' functions. You have to cleverly use other functionality to get the job done."¹⁰⁸

If what you just read sounds rather depressing, take heart and remember that everything should be made as simple as possible, but not simpler.

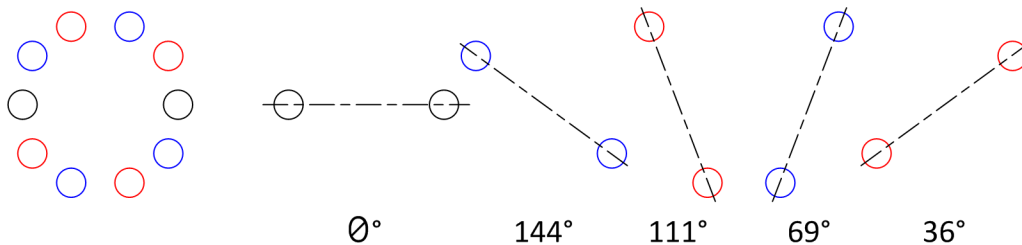
¹⁰⁷ VBA code snippet link: <http://www.textndata.com/forums/how-do-i-array-shapes-147616.html>. More VBA codes can be found at <http://visio.mvps.org/VBA/default.html> if you're interested to explore.

¹⁰⁸ You can find his blog post here: <http://www.visguy.com/2009/11/05/radial-elements-tool/>

With this in mind, here's the breakdown of the metal can IC symbol:



And here's how to create the circular array of pads:



That's right, you simply create a two-circle array spaced appropriately apart and grouped, then duplicate another four similar objects with different angle of inclinations as shown, and finally select all the five objects and perform an [Align Middle-Align Center](#) and group them together.

MOBBED!

Well, not by a flash mob of course, but by components that are squeezed into a certain area of the PCB's footprint. This is more common with analog PCBs, specifically power supplies which need to deliver high power wattages with small form factors. An example is the 350W 0950-9078 power supply that is used in Hewlett-Packard's 748 workstations. There are two sub-assemblies within the casing, one of which is the mains and primary side of the switch mode power circuits, and the other more populous and congested, is the secondary side with the power outputs.

Just drawing the component layout of this sub-assembly is a challenging undertaking, considering the way those components are arranged and mounted which not only makes accessing them difficult, but labeling them clearly in their original symbol layouts as well. How I go about solving it is illustrated in the next two pages—the first, a normal PCB layout with only IC references, and the second, a simplified outline version to facilitate component reference designators and extruded labeling for some of the power points according to their wire colors.

The idea here is to show you an alternative method which you can use in your own work. Incidentally, there are manufacturers who employ this style for their PCB layout diagrams, especially digital designs with high component counts on both sides of the board, in their technical and servicing manuals.

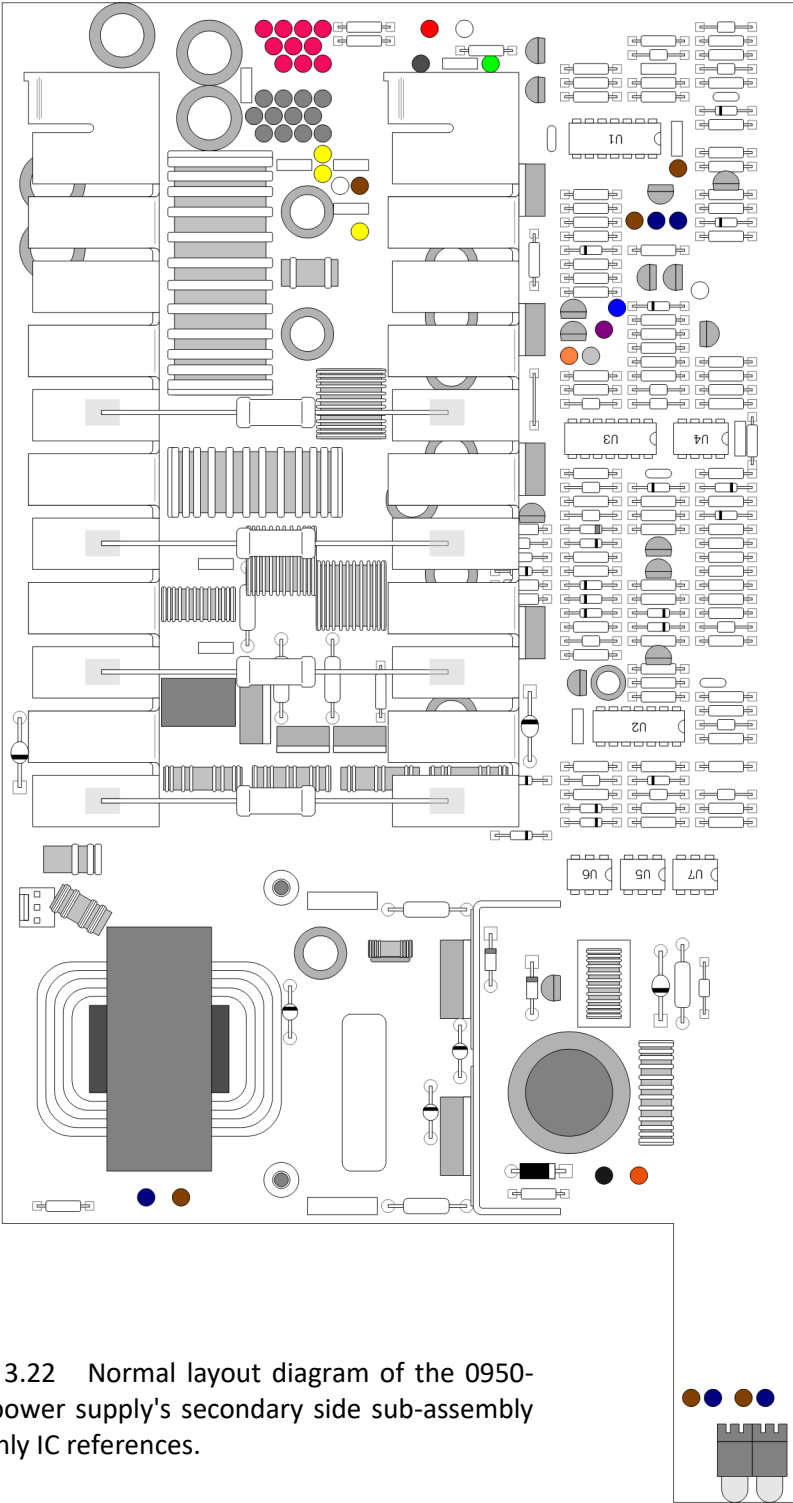


Figure 3.22 Normal layout diagram of the 0950-9078 power supply's secondary side sub-assembly with only IC references.

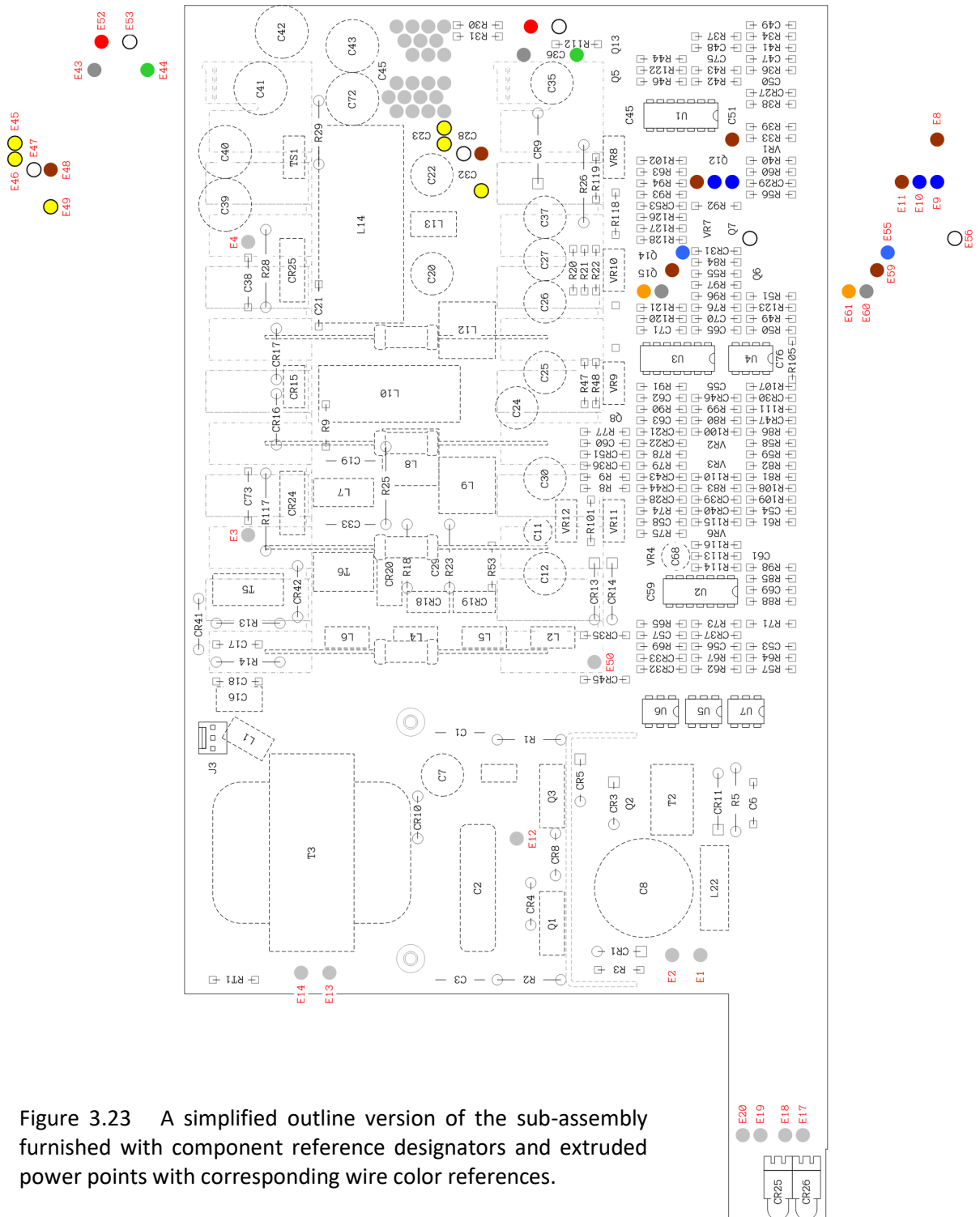


Figure 3.23 A simplified outline version of the sub-assembly furnished with component reference designators and extruded power points with corresponding wire color references.

TOO SMALL TO IGNORE

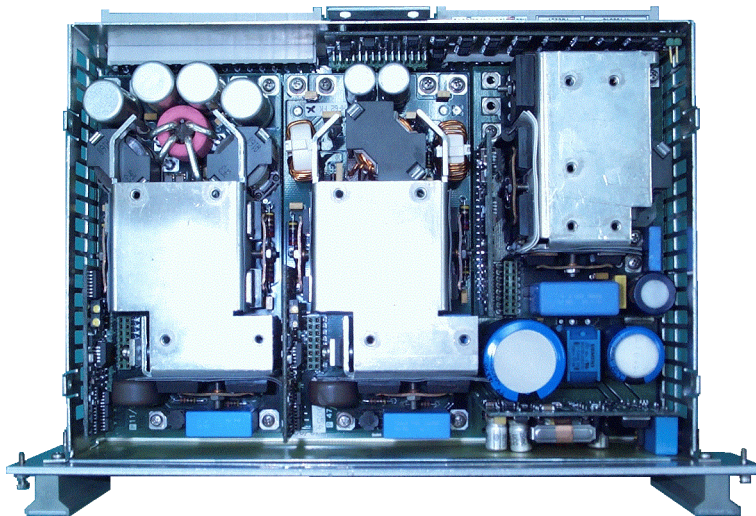
There's one more topic we need to talk about before we close this chapter, and that's dealing with the scale of PCB drawings. Generally, we try to draw the layout of the PCB and its components at 100% or 1:1 scale, if possible—not only to make it easier for measurements and placement, but to ensure that we are drawing correctly based on the physical board.

Most through-hole PCBs should not pose much of a problem at this scale, but it can be quite a challenge as PCBs get smaller and denser with miniature SMT components mounted in close proximity to each other, especially if you want to identify these parts. In such cases, you can do one of two things:

- First, draw the PCB layout at 1:1 scale, then enlarge it sufficiently so you can assign reference designators to the components.
- Similarly, draw the PCB layout at 1:1 scale, then designate only those portions which need to be enlarge and create scaled up copies of them.

The first method is self-explanatory. Overleaf is an example of a logic control circuit card that illustrates what the second method is all about. Each method has its pros and cons; I'll not list them here but let you figure out by trying both methods to better able to decide which works for you.

And that wraps up what we have to say about laying out the PCB artwork. Before we move on to the next chapter, here's an interesting candidate as food for thought:



Think it's possible to draw the layout? Before you hastily say 'No', remember Einstein...

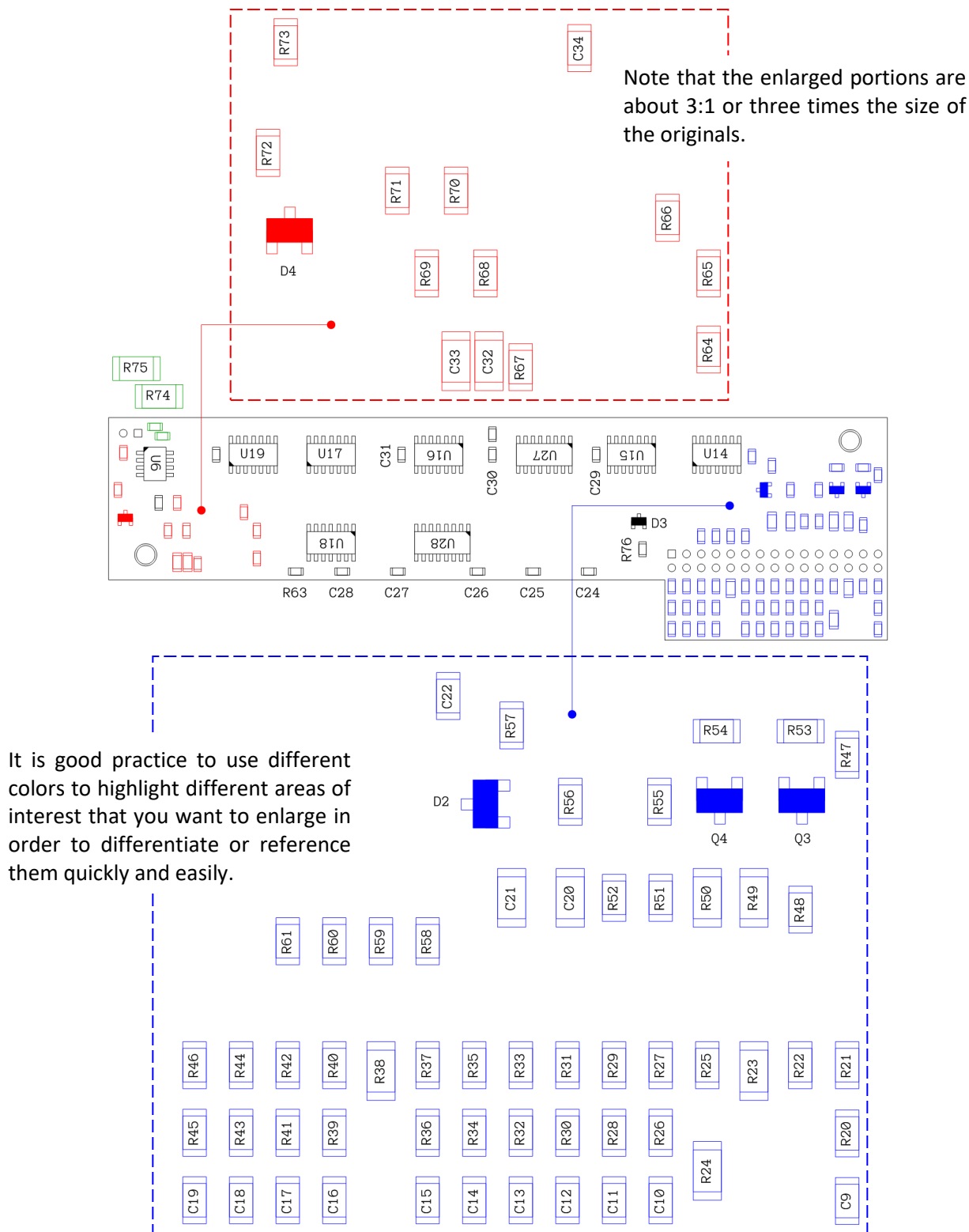
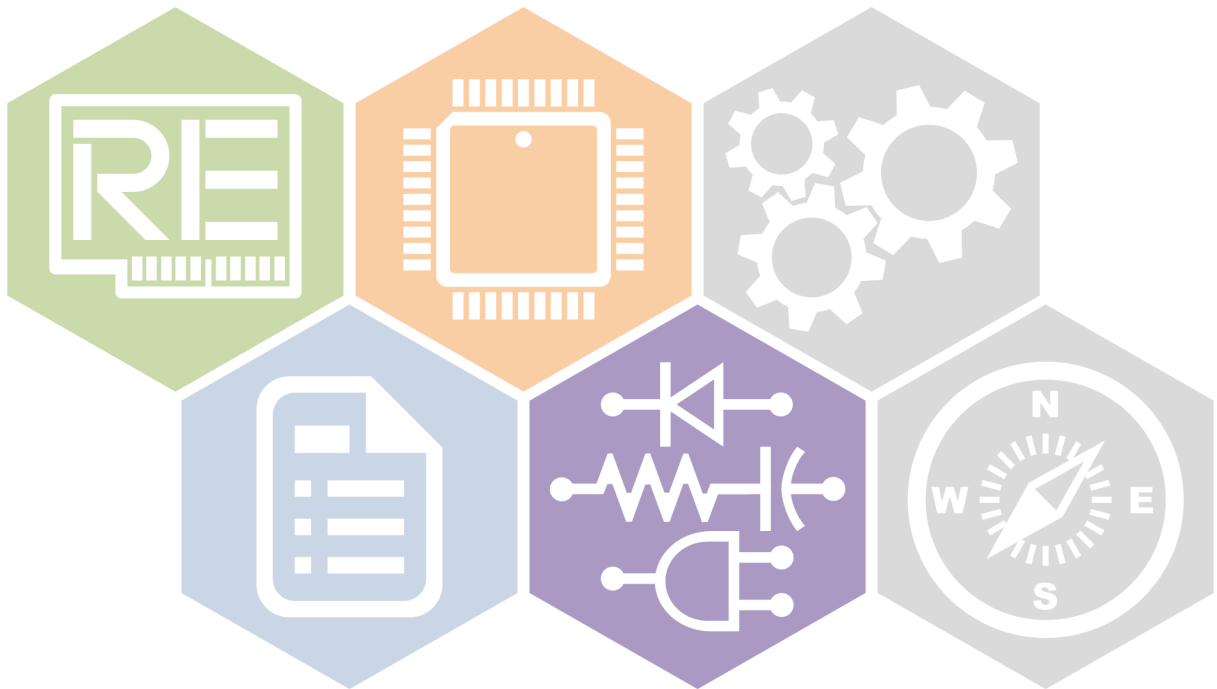


Figure 3.24 Scaling portions of a PCB that is too small for displaying references.

4

Wiring Up The Schematics

**Don't be surprised if you find yourself
in a big tangled mess—
it's normal and to be expected...**



CONTENT

Spaghetti, Anyone?–136

Basic Tools–136

Multimeter–137 Probe Tips–137 Magnifier Lamp–138 Assorted Hand Tools–138

IC Extraction Tools–139 Soldering and De-soldering Station–139

ANSI or IEC?–141

Color or Monochrome?–142

Hierarchical or Flat?–144

Elements of a Good Schematic Diagram–145

Page Sizes and Orientations–146 Reference Designators and Their Placement–146

Circuit Layout and Signal Flow–147 Text and Labels–148 Dots and Crosses–149

Other Considerations–149 Summarizing and Recap–150

Drawing Schematic Symbols–151

Base Reference Grid–152 Logic Gates–152 Tri-State Logic–154 Flip-Flops–155

Passive Discretes–157 Power and Ground–158 Semiconductors–159 Relays–161

Op Amps–162 ICs–164 Buses–170 Signals and Labels–174

Approach and Strategy–175

1. Validate all power and ground connections–179

2. Validate address and data bus–180

3. Validate known signals–182

4. Validate jumper connections–183

5. Validate visible traces–184

6. Unknown and hidden traces–184

7. Unused pins–185

It's an Analog World!–188

An ATX Power Supply Example–189

Strategy for Analog PCBs–192

1. Validate all power and ground connections–192

2. Validate low impedance paths and parts–192

3. Validate known configurations–193

4. Validate visible traces–196

5. Unknown and hidden traces–196

One Useful Trick–197

The Magic Carpet Method–197 The Gold Finger Method–198

A Word of Caution–198

Doing the Unthinkable–199

4

WIRING UP THE SCHEMATICS

"I've failed over and over and over again in my life... and that is why I succeed."

Michael Jordan

Without dispute, Michael Jordan¹⁰⁹ is an NBA superstar and legend, not just for his amazing basketball skill and mastery of the game, but his ability to consistently deliver the scores in his NBA careers despite coming back from two retirements. In a documentary about his life, he attributed his success primarily to one thing—hard work. Every day without fail, he would devote hours perfecting his three-point shooting¹¹⁰ and refining his slam dunk techniques.

Another legend that earned my admiration and respect was the late martial arts exponent, Bruce Lee (1940-1973). I've watched a number of his kung fu movies as a boy, such as *The Big Boss*, *Fist of Fury*, and *Way of the Dragon* (co-starring Chuck Norris). Ask a Chinese to name one famous martial artist and Bruce Lee's name invariably gets mentioned, despite being dead for almost four decades. The arduous training regime he subjected himself to surpassed many practicing masters and instructors even to this day.¹¹¹ And rightly so, as he once resolutely remarked, "Do not pray for an easy life, pray for the strength to endure a difficult one."

So what has that got to do with tracing out the PCB's schematic diagram? Much in every way, just like learning a skill and perfecting it. If you really want to do something worthwhile, might as well give your very best shot or don't do it at all. And being passionate about what you want to do will certainly make a big difference, not to mention the satisfaction that follows.

¹⁰⁹ Incidentally, he was born in the same year as me, though I'm slightly over one and a half months older than he is. Destiny though, dictate that we go different paths and achieve different things in life.

¹¹⁰ It is said that he practiced at least 3,000 shots each day during his active career with the Chicago Bulls. Is it any wonder he became the only player besides Wilt Chamberlain to score 3,000 points in a season, averaging a league high of 37.1 points on 48.2% shooting, not counting his record 200 steals and 100 blocks in one season?

¹¹¹ Before Sylvester Stallone came on the silver screen to impress the audience with his one-arm push-ups in his role as Rocky, Bruce Lee already demonstrated it in an interview in front of the TV—using not one hand but just two fingers!

SPAGHETTI, ANYONE?

Like many established companies in Singapore, my company regularly accepts requests from tertiary education institutes to host their students as interns¹¹² during their school holidays, to allow them to gain some industrial exposure and experience the real work life.

During one of the in-takes several years ago, I was assigned two second-year polytechnic students, both enthusiastic and eager to learn. I tested them on some basics in electronics and asked if they knew or had drawn schematic diagrams before. They told me the only source they come across are from the text books and some sketches they made in their lecture notes.

Not too bad, I thought. So after giving them a little tutorial on Visio, I handed them a simple logic board and asked them to do a reverse engineer on it. Two weeks later, they came back to me with an A3 print-out of the board's schematic diagram, which resembled a spaghetti artwork of crisscrossing wires around logic symbols and various IC blocks that are disproportioned and displaced.¹¹³

"That's a good first attempt. It can use some facelift though..." I tried to sound positive and encouraging, and then followed through with a short two-day course on reverse-engineering and schematic drawing with them. After that it's back to the drawing board for a week and it proved to be a big improvement from their initial effort, though it definitely needed further refinements to achieve a respectable print. So it is with each PCB you'll be working on, that you'll make further in-roads and improvements with your skill and styles. As Winston Churchill (1874-1965) once said, "Success is not final, failure is not fatal: it is the courage to continue that counts."

BASIC TOOLS

First things first. Depending on the kind of PCB you are reverse engineering, you'll need some or all of the following tools:

- Multimeter and probe tips
- Magnifier lamp
- Assorted hand tools
- IC extraction tools
- Soldering and de-soldering station

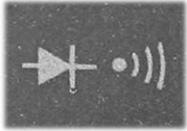
Some of the above items may seem a bit strange to you at first, but you'll understand their uses soon after. Notice that I did not include PCB layout diagram, parts list, datasheets and even the highlighters here—you should have them ready by now.

¹¹² A student or a recent (fresh) graduate undergoing supervised practical training. It's also termed industrial attachment or IA in short.

¹¹³ It would seem like a pity I did not keep a copy of their before and after handiwork to let my readers see the contrast. On hindsight, I'm glad I didn't—I certainly do not wish to embarrass them unnecessarily with their once amateurish works, now that they've graduated and joined the ranks of fledging engineers.

Multimeter

The multimeter is a compulsory gadget, in fact the main tool that you'll depend heavily on for PCB reverse engineering work. While you can use an analog multimeter to do the job, my advice is to get a digital one, a digital multimeter (DMM) with continuity measurement function. I personally use the Fluke 77 model shown on the right for its simplicity and reliability, though any DMM with an audible beep tone¹¹⁴ function will do just fine.



Some DMM has the diode function and continuity mode separately while other models combine both, such as the Fluke 77, to make it easier to use.

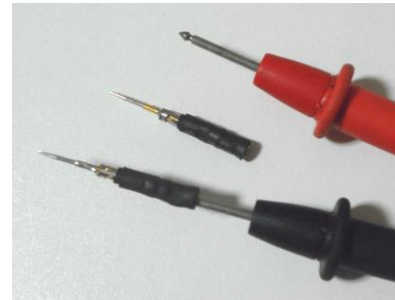


The reason for using a DMM with a continuity function is obvious—it enables you to quickly probe for continuity without having to look at the meter reading each time, which not only saves you time but also from an aching neck! However, bear in mind that the continuity mode not only sounds the buzzer for short but low resistance as well, so it is important to momentarily glance at the meter reading whenever the buzzer sounds.

The challenge, of course, comes from working on a PCB that has many low resistance components such as inductors, small value resistors (<100) and transformers, etc. That's why it's important to draft out a parts list for the PCB, so you can take note of these components and make the necessary arrangements (such as lifting a lead for through-hole parts, or removing for SMT parts) so they do not interfere or lead you down the wrong track (pun intended).

Probe Tips

The probes that come with the DMM are usually good enough for PCBs that have been stripped of conformal coatings. Sometimes, the coating may be tough to remove (such as Paralyene), or you may just want to do a preliminary probe on certain area of the PCB. In such instances, it'll be helpful to have probes with extra sharp tips to easily piece through the coating without tiring your wrists due to strenuous exertion, and to prevent deep pock marks all over the solder pads.



These sharp probe tips can be easily made from leftover in-circuit test (ICT) probes (or even steel nails or needles) soldered to one end of a crimp socket and heat-shrink sleeved for added grip and insulation, and the other end of the crimp socket for mating with the DMM probe tip.

¹¹⁴ I have a colleague who has hearing problem who simply could not hear the beep tone when using the diode function. For those of my readers who may have similar impairment, I would suggest getting a DMM that has both audio and visual indication for continuity measurement, such as the MOTWANE M21C model or the Agilent U1230 series DMMs.

Magnifier Lamp

The constant strain we put our eyes through working on PCBs with increasingly miniaturized SMT components and board densities will eventually cause our eyesight to deteriorate overtime. A magnifier lamp will greatly ease the eye strain by magnifying and illuminating the PCB. I highly recommend the LT-86C model (my workshop has a couple of these). And while you're at it, do yourself a favor and get a handheld magnifier for added measures.



Assorted Hand Tools

Common hand tools are good to have around, such as those listed in the figures below:

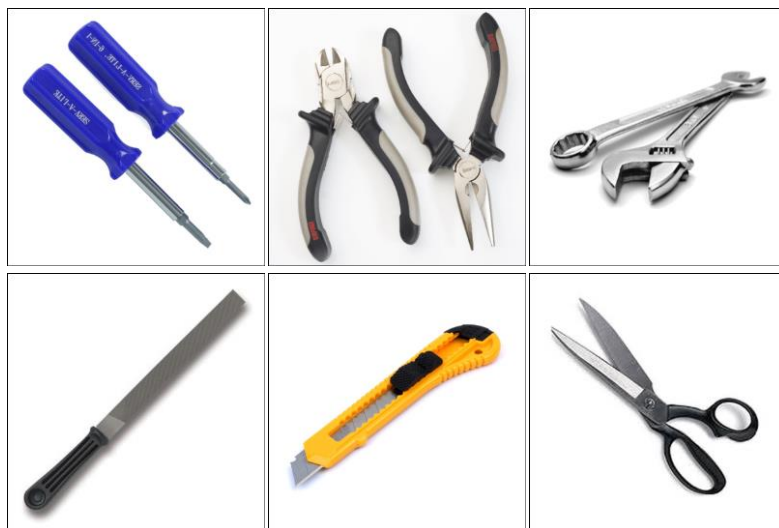


Figure 4.1 Assorted Hand Tools: (a) Screwdrivers, (b) Pliers and cutter, (c) Spanners, (d) Mini hand file, (e) Pen knife, and (f) Scissors.

Not all PCBs are just simple boards populated with components. Some may come with additional daughter boards (like the MVME 166-14A with a mezzanine memory card) that are fitted and fastened with screws on the main boards. Some may have protective fittings over fragile components or which act as heatsinks that are bolted. Still, others may have plastic tie-wraps or waxed threads securing ICs that are socket-mounted. In such situations, you'll need hand tools to loosen the screws or bolts in order to separate the daughter boards from the main boards, remove fittings for better accessibility, or even cut off the tie-wraps or securing threads to remove ICs from their sockets. The hand file and pen knife do come in handy in instances where the conformal coating are difficult or impossible to remove, in which case you'll have to resort to scrapping or filing off the coating from the tips of the component leads to establish electrical contacts.

IC Extractor Tools

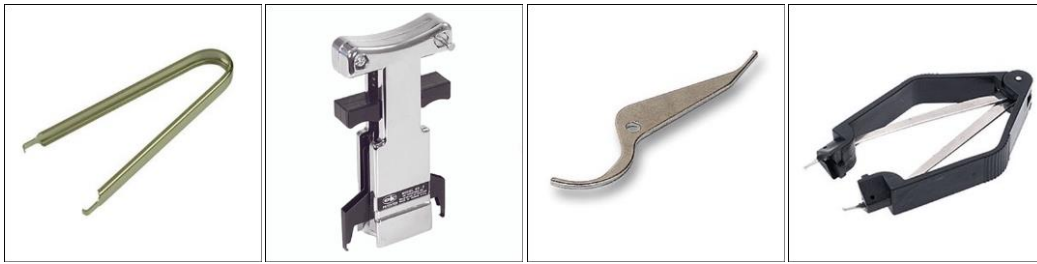


Figure 4.2 Types of IC extractor tools: (a)-(b) DIP, and (c)-(d) PLCC.

Why the need to remove ICs from their sockets? For one thing, there could be SMT components hidden below them which you need access to, or you may need to read the content of the ICs to help you with your work. While you may use either a sharp pincer or a slim flat head screwdriver to lift an IC out of its socket, it may not be such a good idea as using other components for a fulcrum point may inadvertently dent, deform or damage them if the force exerted on them becomes excessive.¹¹⁵ IC extractor tools are meant to do the job, and do it safe and well. Figure 4.2 shows both the simple and sophisticated forms of extractor tools for the DIP and PLCC types of ICs. I have used all of them at some point in my work, and would advise my reader to go for the better quality ones as they're less likely to cause accidental damage to the IC, and in the case of the PLCC, the socket also.

Soldering and De-soldering Station



Figure 4.3 Soldering and de-soldering station (courtesy of PACE Incorporated)

¹¹⁵ At the time of this writing, I've encountered two VME PCBs with damaged pins on the SMT ICs located below DIP socketed RTC/NVRAM modules, which is the result of using screwdriver to lift them from their sockets instead of using proper IC extractor tool.

Along the way it might become necessary to partially or completely remove soldered components to ascertain the connectivity of hard to access probe points, or if the components are interfering with the validation process. That's when you'll need to resort to using soldering and de-soldering tools.

During my polytechnic days I've used a 35W soldering iron that plugs directly to the wall mains, and a rather primitive de-soldering pump that's hand operated. It's only when I worked in the air force as a technician that I was trained to use the PACE re-work station.¹¹⁶ While a unit comprising three channels that support a solder tip, a vacuum-activated extraction tip, and a pincer SMT removal tip, may cost a few thousand dollars, it's still a good investment. My workplace alone has three of these MBT 250 models. There are less expensive alternatives but none of them come close to the ease and efficiency of PACE's patented vacuum-activated extraction technology.

However, if you're on a budget, I would recommend Xytronic's XY-D1-3 30W vacuum pump de-soldering iron which has a good quality manual solder sucker and heating element. It also comes with XY-T-1 0.59" tip which resembles the PACE's extraction tip (see inset).

The advantage of this tool is it frees you from having to hold an iron and a vacuum pump at the same time. At less than \$50 a piece, that's a pretty good deal.



We've covered quite a number of basic tools that may be required in the course of your reverse engineering work, though not all of them unless you work with a wide variety of PCBs. Of course, some of my readers may suggest a few more based on their experiences, which I think is fine, but which I don't intend to include more than the bare essentials here. However, I will make mention of one other necessity later on when I touch on the practical aspect. For now, allow me to keep you in suspense just to keep you on the edge of your seat...

As we enter the lesson proper, there are a number of questions that need to be answered to get you on the right footing:

- Which type of symbols do I want to use for my schematic diagrams?
- How do I make proper use of colors?
- Should I employ hierarchical or flat organization for multi-sheet designs?

Let's look at each in turn now.

¹¹⁶ For those of you who've had the privilege of being trained on the PACE machine under a certified PACE instructor, you'll know the arduous repetitive drills you're subjected to—from stripping and tinning multi-core wires, soldering the wires to different kinds of terminals, soldering discrete and IC components, repairing broken tracks and restoring a multi-layered PCB! Solder for inspectability sounds easy but takes lots of practice to achieve; however, once you've gone through the baptism of fire (of the soldering iron, that is) you'd no doubt be imbued with a valuable skill that sets you apart from the rest!

ANSI OR IEC?

If you've read enough electronics books and magazines, work on PCB design projects, or repair in-house or third-party PCB products, you'd most likely have come across schematic diagrams with different type of circuit representations, some of which may even get you scratching your head just to make some sense out of them. Welcome to the real world!

Consider the eight basic logic gates and their symbolic representations below:

Gate	ANSI/MIL	IEC	DIN	Gate	ANSI/MIL	IEC	DIN
BUF				OR			
NOT				NOR			
AND				XOR			
NAND				XNOR			

Figure 4.4 Logic gates and their symbols in ANSI, IEC and DIN representations.

For many of us, the first column representation is familiar like an old friend. This type of logic symbols is known as distinctive shapes and are commonly found in more traditional or simple schematic diagrams. It has its origin in the 1950s and 1960s from the US military under the MIL-STD-806 specifications drafted to standardize all electronics drawing documents. It is a 'Made in America' thing which is why this style of drawing is also known as the ANSI/MIL standard.¹¹⁷

The IEC¹¹⁸ style of representation, as can be seen from the middle column, is a rectangular shape with logic notations to describe a device's functionality. US engineers who've come across this style found it rather 'unfriendly' compared to the ANSI style, but it is widely adopted by many European countries such as UK and Germany, the powerhouses of electronics innovation and design. The reason is because the IEC standard provides a consistent method of describing complex logic functions for digital circuits than is possible using the ANSI standard. Engineers who are comfortable with mathematical symbol notations will appreciate the elegance of this style, but those who don't will probably irk it.

The last column is known as the DIN or Deutsches Institut für Normung, which translated is German Institute for Standardization. While it is not as common as the ANSI or IEC, some countries in Europe still use them.¹¹⁹

¹¹⁷ ANSI stands for American National Standards Institute, and MIL stands for military.

¹¹⁸ IEC stands for International Electrotechnical Commission, and is a non-profit, non-governmental international standards organization that prepares and publishes international standards for all electrical, electronic and related technologies—collectively known as 'electrotechnology'. (Source: Wikipedia)

¹¹⁹ They are such rare gems nowadays—but I had the honor of encountering some of these in my work!

While I'm tempted to give equal treatment to the ANSI and IEC standards, I'll have to limit myself to just using the ANSI style for all my examples in this chapter, again for obvious reasons. For those interested in the IEC standard, there is a booklet by Texas Instruments which gives an overview of the IEEE Standard 91-1984 of the same title. Just google to find and download it.

COLOR OR MONOCHROME?

Color is a powerful medium that can enhance the overall appearance of a drawing, be it PCB layout or schematic diagram, if use appropriately and moderately. Those who have worked with CAE or computer aided engineering software know that these tools provide a rich set of color schemes that are not only customizable to the smallest detail, but can also be overwhelmingly complex to manage to achieve a well-balanced visual effect. Finding that illusive blend is difficult, because every person has a different perception and appreciation of colors.

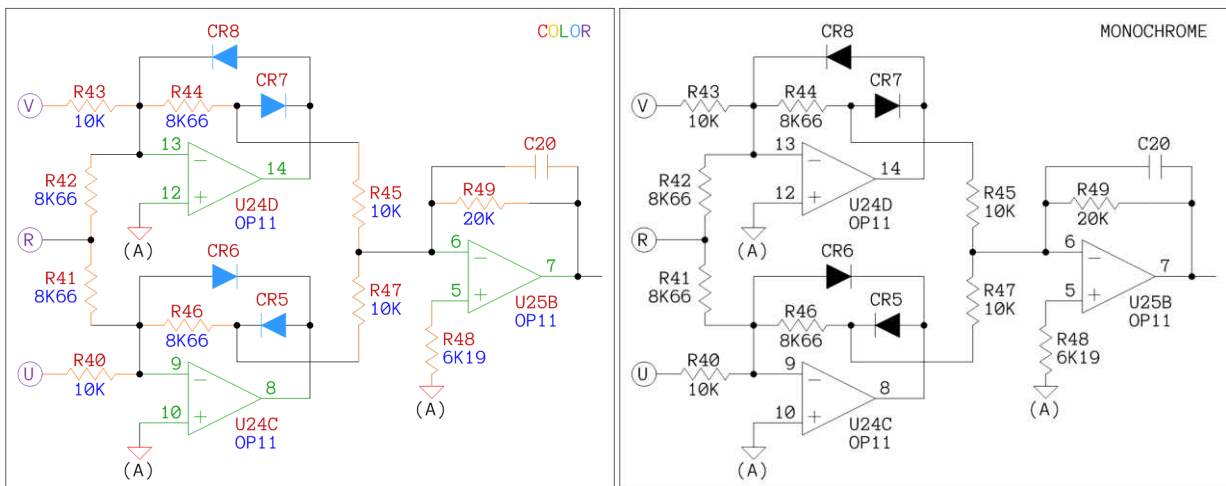


Figure 4.5 Schematic diagram in color (left) and monochrome (right).

Here are two samples of the same circuit, with and without colors. The one with colors certainly looks more exciting and appealing compared to the one without. The different color schemes for different component types make them stand out and easily distinguishable as well. So it seems that using color is the natural thing to do—except that you need to ask yourself the following two questions:

- Will I be printing in color (is it even necessary)?
- Am I willing to spend more time to manage the color schemes for overall consistency?

Granted, color laser printers are getting cheaper nowadays but one capable of printing A3 size papers still costs over \$1,000 at the time of writing. So unless you want to limit your drawings to only A4 sizes, which means having to break up your schematic diagrams into more sheets, and in the process makes the circuit harder to understand because related logic and functional connectivity cannot be contained within the same page, retaining the A3 size advantage is not an option but necessity.

Another challenge is assigning color schemes to the myriads of component types, such that different shades of a color may become indistinguishable when symbols that use them are placed together in one sheet. This can be frustrating and self-defeating—besides having to focus on the reverse engineering task, you'll need to keep tabs of the use of colors—an additional burden that you'll come to realize that you can gladly do without.

So should we just forget about colors altogether? Well, no. Like I said earlier, colors do have their uses but you need to know when to put them to good use to achieve the best effects. Personally I draw all my schematic diagrams in monochrome, and use certain colors for certain purposes. Take a look at part of a video processor board for example:

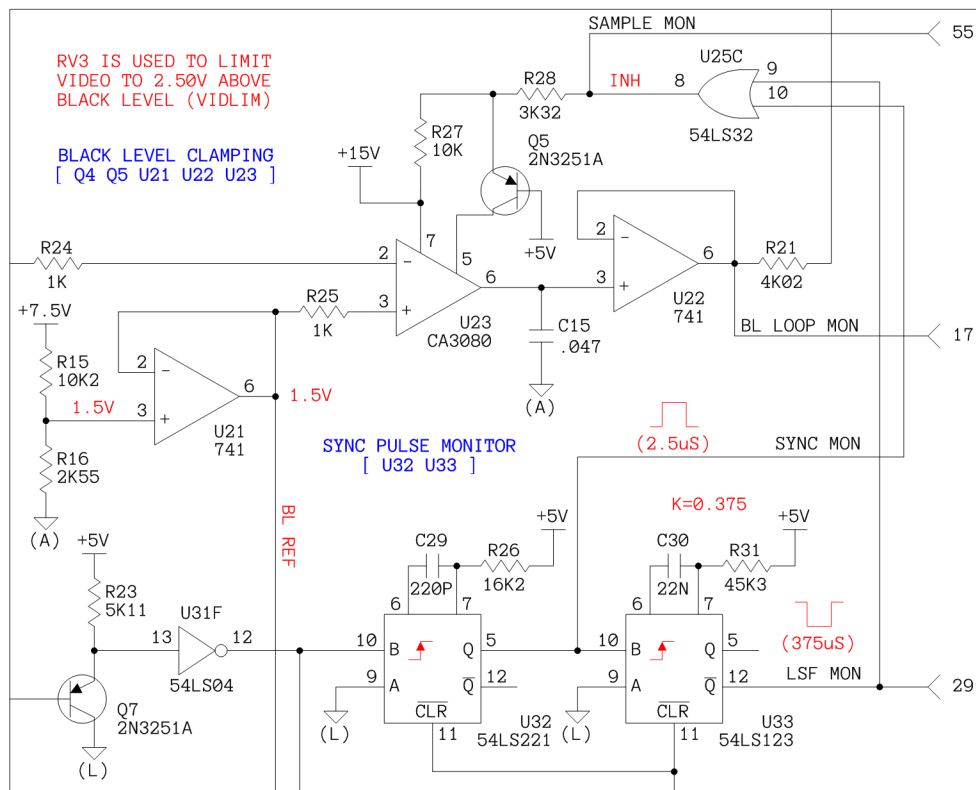


Figure 4.6 Proper use of colors can make a schematic diagram clearer.

As you can see, I use **blue** to label circuit functions and **red** for a couple of purposes, such as denoting voltage and pulse width values of interest, distinguish critical signal names from common ones, and even comments for some component's usage and purposes.

So the choice is not between color or monochrome, but when to use colors as necessary. If you grasp the underlying principles of what makes a schematic diagram more readable and understandable, you are on your way to creating great schematics that others can appreciate!

HIERARCHICAL OR FLAT?

As with PCB layout, drawing a schematic diagram also requires some planning before you start. One of the consideration is the layout of schematic drawings that span multiple sheets or pages. I've drawn schematic diagrams that fit nicely into one single A4 size paper, but these are usually simple circuits of small PCBs or power supply modules. More often than not, if you have a PCB of medium complexity i.e. containing an average of 50-75 SSI¹²⁰ type ICs—you may well be looking at a minimum of three A3 size sheets or more, depending on how you optimize and organize the placement of the component symbols and wire networks.

A hierarchical layout is a top-down approach that allows many levels or layers of nesting, progressing from simple to complex representation. Unless you are doing serious PCB design involving large amount of components, it is quite unlikely that you'll need to go beyond two levels of abstraction. A flat layout, on the other hand, is a straightforward, single layered approach in which the various sheets are inter-linked relationally and laterally (side-to-side).

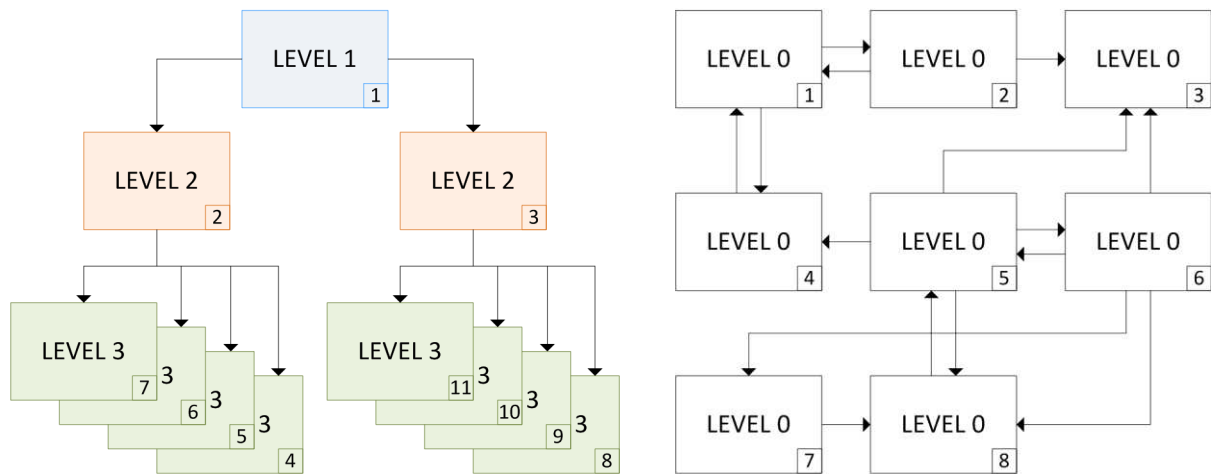


Figure 4.7 Hierarchical versus flat schematic diagram models.

The figure above shows the visual concept of these two models. It must be understood that for the flat model, the level zero implies that there is no level but only a single layer of representation, while the arrows are indicative of the cross-border circuit interconnectivities between sheets. For schematic diagrams spanning more than ten pages, a hierarchical layout is better because its top-down organization makes it easier to search for circuit portions based on block diagram functionalities. I have worked with OEM's schematic diagrams that have over 30 sheets while developing test programs for those PCBs, and I can honestly say that without a hierarchical organization, it would be a nightmare just trying to find my way through the schematic labyrinth!

¹²⁰ This is just an estimation since a PCB rarely consists of only SSI (small-scale integration) ICs but a good mix of other peripheral (MSI, LSI) and processor (VLSI) ICs as well. You may roughly equate four SSI to one MSI, four MSI to one LSI, and four LSI to one VLSI, etc. but that again is subjective to the type and functionality which are simply too complicated to breakdown accurately.

Thankfully, in all my years of doing reverse engineering, the most complicated PCB I've ever completed only managed a maximum of six A3 sheets, with the uncompleted ones potentially coming close to ten. What this means is that there's a trade-off to doing reverse engineering compared to doing PCB design—the former, by its very nature, is often a one-man effort and therefore is very much limited to a certain complexity that the engineer can handle; the latter, however, can be a team effort in which different engineers are assigned specific parts of the design they are proficient in, aided by the ever sophisticated and powerful design tools available, thereby achieving complexity levels never before dream of.

Base on personal observations and experience, it is my humble opinion that the flat model is more suited for PCB reverse engineering compared to the hierarchical model, simply because the scale of work points to it and at the same time it'll keep your sanity intact, probably.

ELEMENTS OF A GOOD SCHEMATIC DIAGRAM

So what makes a schematic a good schematic? I'm sure in the course of your work you've come across many kinds of schematic diagrams, some of which you enjoyed viewing or analyzing, while there are those you wished you never had to refer to.

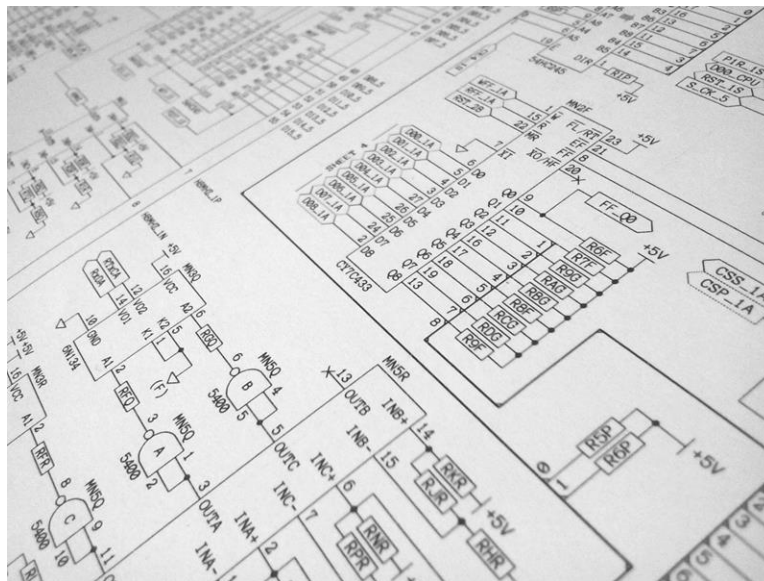


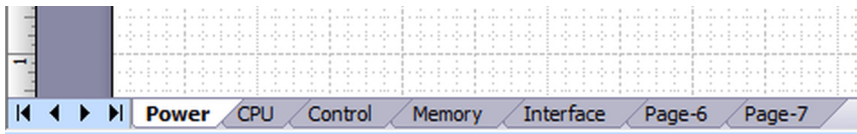
Figure 4.8 Consistency of style and good readability.

Spotting a good schematic diagram isn't that hard, but drawing one yourself is if you have no idea what those essential elements are. It can really mess up your work after you're done drawing you probably wouldn't want to look at it again... Well, take heart! Drawing great schematics can be second nature if you take heed of the following guidelines that I'm going to discuss—the secret to bear in mind is this: [consistency of style](#) and [good readability](#). It's really that simple.

1. Page Sizes and Orientations

Unless you are into blueprint and architectural drafting business, most likely you don't have the luxury of a Graphtec roller plotter that allows you to print up to A0 size papers. Being able to draw on page sizes larger than A3 does not necessarily translate to clearer schematics. It can be much harder to manage with all the endless scrolling and panning, not to mention the problem of getting hardcopy printouts.

Though Visio has a rich set of paper sizes to choose from, it'll be good to just settle on two or three common sizes such as the A3 and A4. I have a 19-inch widescreen LCD monitor with 1440 x 900 screen resolution, and an A4 or letter size landscape drawing fits quite nicely at 83% for an overall view, while an A3 or ledger size landscape drawing will have a 58% page view.¹²¹ In terms of page orientation, landscape is usually the way to go since it can better accommodate the general flow of signals and circuits horizontally than vertically. There are exceptions, of course, such as the direct point to point connections found on backplanes and interface circuits.



As your drawing unfolds, you may want to organize them into related clusters based on their functions, such as power, CPU, control, memory, and interface sections, etc. Like Excel, the Visio page tabs below the active drawing page allow you to rename them from the default Page-1, Page-2... format to your own custom named pages so you know which page to navigate to your desired circuits.

2. Reference Designators and Their Placement

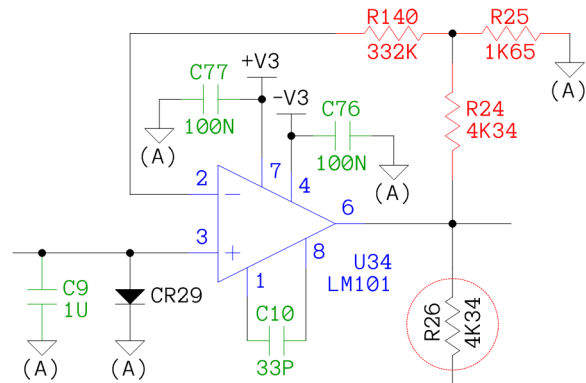
The choice of reference designators is so obviously important that sometimes it's taken for granted or overlooked. Unless the PCB you're working on uses odd reference designators,¹²² it's best to keep to the standard prefixes such as those listed on [page 45](#).

Placement of component designators and values are equally important, not just for aesthetical reason but to prevent clutter or overlap that not only makes it hard to read the information, it could even lead to wrong interpretation of component values.

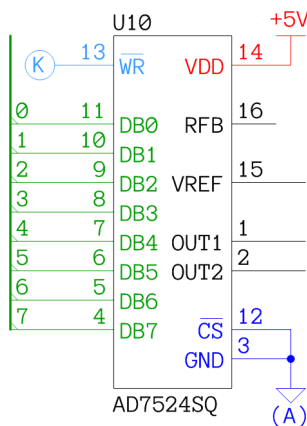
¹²¹ My office PC uses a 17-inch 4:3 LCD monitor with a 1280 x 1024 resolution, so an A4 landscape shows up at 100% and an A3 landscape at 67% page view. Note that if you're using a version of Visio with the ribbon feature, the available page view will be much smaller.

¹²² Eastern European countries seem to favor different kinds of component prefixes compared to their western counterparts. For example, ICs are designated as AR, MN instead of U, IC and resistor networks as Z instead of RN, etc. Maybe it has something to do with their language and spelling, I guess.

Consider the circuit on the right: The resistors (red) and capacitors (green) use different placements for reference designators and values for horizontal and vertical orientations. Occasionally, space constraint may require the use vertical texts (see circled resistor), but be careful not to use them too liberally as it can pose difficulty for some people to read your schematic.¹²³



3. Circuit Layout and Signal Flow



The standard practice is to have positive voltages (red) on top and negative voltages or ground (blue) at the bottom,¹²⁴ preferably with signals that are tied to these power sources grouped together with them, if possible.

Signals or logical data lines should also flow from left to right, with exceptions of feedback and common logic control signals which may link to multiple devices. Component pins should also be placed or grouped according to their functions and not according to how they appear on the datasheets. If possible, use direct connections between components to reduce wire crossings which may clutter and waste otherwise useful space that other components can occupy. For group signals like address and data, you should employ bus lines (green) to organize them for better clarity.¹²⁵ Bus lines are double the thickness of a signal line.

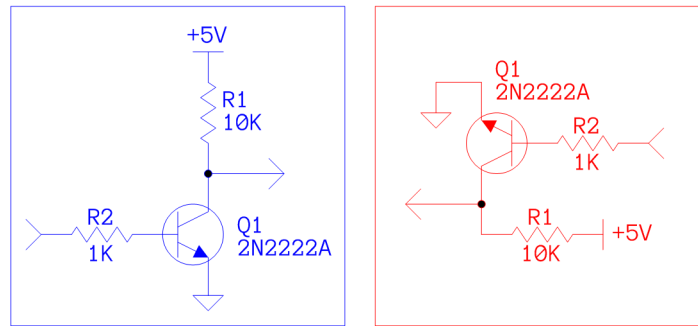
ICs by their very nature, are more straightforward to determine in terms of their pin placement.¹²⁶ Discrete or analog components such as transistors and MOSFETs, however, can be more tricky to gauge or orientate, so you may need to re-arrange them and their surrounding components to make the circuit more readable. For example, consider a simple transistor switch circuit (see overleaf)—which is easier to identify from one look, the left (blue) or the right (red) figure? Remember, a good schematic shows you the circuit, but a bad one makes you decipher and decode it.

¹²³ I don't discourage using vertical texts in schematic diagrams since I use them myself, but make sure there is good reason for their use or you might end up with a drawing that's awkward at best or a pain to behold at its worse!

¹²⁴ Sometimes space constraint may not permit it, like U34 in the diagram with both positive and negative voltages placed on top to give way to compensation capacitor C10 below. Flexibility must be matched by common sense.

¹²⁵ Imagine having to individually route the 20-bit address and 16-bit data buses of a CPU to the various support ICs (buffers, EPROM, RAM, peripherals, etc.). It's gonna be worse than the nightmare on Elm's Street!

¹²⁶ There are exceptions, of course, such as op amps and analog switches. These will be discussed in their relevant sections later on.



This brings out one important point in reverse engineering—if there's anything you can expect, it is a constant need to change, to re-orientate your partially completed circuits as they unfold and offer more clue as to their purposes and functionalities, sometime in small proportions while other instances it may even be necessary to do a major overhaul. With practice and the right strategies, though, you can keep this to a manageable minimum.¹²⁷

4. Text and Labels

Take a look at Figure 4.6 again to see how text and labels are used beside reference designators. You'll notice that they are all in uppercase letters. That's right! Strange as it may be, it has been proven that engineering drawings look better and more readable when text and labels are in capital letters. It could be due to the fact that monospaced fonts appear better in caps than lowercase.

When drawing large schematics, it is also good practice to label key or recurring signals that are found on separate sheets, such as address, data and control lines. Besides making it easier to trace the signal connections, it gives the schematic diagram a clearer and more organized look. A point to note: keep the names short but reasonable enough to make sense of their meaning. For example:

Addresses	A0, A1, A2 ... A15
Data	D0, D1, D2 ... D15
Control	RD, WR, CS, OE, EN, DIR, RST, CLK, INH, INTR
Power ¹²⁸	VCC, GND, VDD, VSS

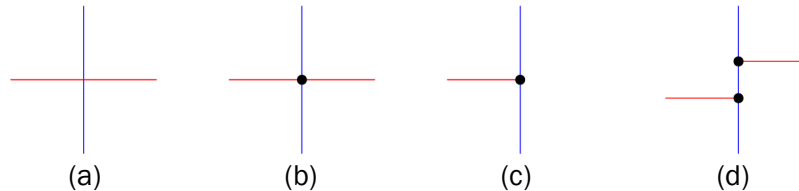
The general rule of thumb is to limit to no more than five or six characters, especially signal names that are within the confines of IC symbols. Exception applies on labels for signal lines and connector ends, if clarity is a necessity.

¹²⁷ Reverse engineering is really like doing jigsaw puzzles. Finding the pieces that fit and putting them in the right places require more than just determination or enthusiasm; you need good strategies too, and that's what we'll be covering later on in this chapter.

¹²⁸ You may replace text-based power notations (VCC, GND, VDD, VSS) with number-based (+5V, 0V, +15V, -15V) but be sure that they reflect the true voltage values that are used for that PCB or it will result in disaster when incorrect power is applied.

5. Dots and Crosses

Nope, I'm not talking about Othello or the ancient game of Go here. I'm talking about connections—nothing of the French¹²⁹ sort but everything to do with electrical connectivity. A good schematic is all about right connections between components that make logical sense. Period. Get the pun? The general rule that **dots connect but crosses don't** may not fully explain the intricacies of the do's and don'ts of good connectivity, so I've provided some visual examples to illustrate the point:

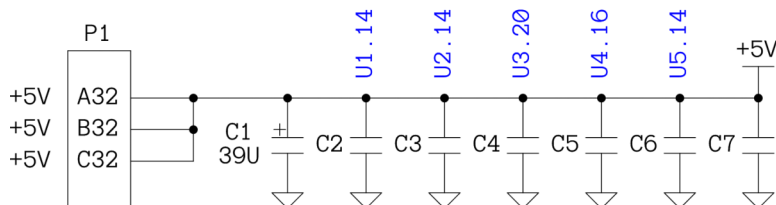
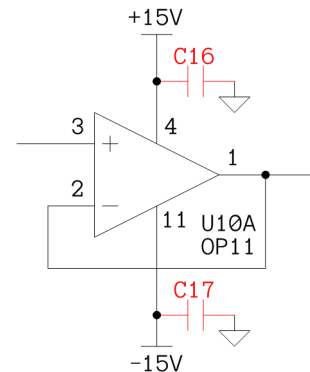


Generally,

- (a) If two wires simply cross each other, it should be understood to have no connection.
- (b) If there is a dot or junction on the point of crossing, the two wires are connected, but try not to use this method, rather use (d) instead.
- (c) If possible, always keep to T junction and keep cross junctions (b) to a minimum.

6. Other Considerations

Some design engineers advocate placing decoupling capacitors close to their ICs. This is good practice and is often used in analog schematics containing parts like the operational amplifier (see inset). But it is not mandatory and does not necessary translate to better schematic diagrams. In fact, having the decoupling capacitors next to their ICs will mean allocating extra space near each IC's power pins; for a PCB with many ICs the space taken up will be substantial without really achieving any useful purpose, and may even make routing wires between ICs more difficult. A neater way would be to group all decoupling capacitors according to their power rails and denote the IC power pins they are connected to, as shown below:



¹²⁹ Admittedly, like the Germans, the French has their powerhouse in the aviation and electronics industries like their home-grown defense company, Thales Group, who design and build systems as well as provide services for the aerospace, defense, transportation and security market sectors. I've had the privilege to work with some of their engineers and systems, and I'm impressed with their capabilities and innovations.

Alternatively, you can denote the IC power pins separately in a tabulated form for quicker and easier reference as shown below:

Table 4.1 IC Power and Ground Pins Reference Table

REF DES	POWER CONNECTION
U1 ,U33,U35,U36	GND=16 VCC=32
U2 ,U14,U20,U32	GND=12 VCC=24
U3	GND=10 VCC=20
U4	GND= 7 VCC=14
U5 ,U12	GND=11 VCC=22
U6	GND= 7 VCC=14
U7	GND= 7 VCC=14
U9 ,U11	GND=10 VCC=20
U10,U21-U23,U26,U28	GND=12 VCC=24
U13	GND=B6,C6,E3,E9,F3,J6 VCC=C5,C7,F9
U15	GND= 7 VCC=14
U17,U18	GND=10 VCC=20
U19,U24	GND=10 VCC=20
U25	GND=B1,K11,L2 VCC=A10,B10,H2,L6
U27	GND=14 VDD=13 VSS= 3
U31,U37	GND=14 VCC=28
U34	GND=1,12,22 VCC=24 VSS= 3
U38,U43	GND=18 VDD=24 VSS=22
U39,U42	GND=12 VCC=24
U44,U66	VDD= 3 VSS=12
U57,U58	VEE= 3 VSS=12
U45,U54	VDD=14 VSS= 8
U46	GND= 7 VCC=14
U47	GND= 7 VCC=14
U48,U50,U51,U55	VDD= 7 VSS= 4
U49,U56	VDD= 7 VSS= 4
U53,U60,U65	VDD= 8 VSS= 4
U61,U62,U63,U64	VDD= 2 VSS= 7

Summarizing and Recap

What is a good schematic diagram? A good schematic diagram is more than just a sheet of nicely drawn circuit with electrical symbols and connections. Beyond the aesthetic appearance, a schematic diagram should exhibit the following characteristics:

1. **Consistency.** Components are the heart of a schematic diagram, and maintaining a consistent style of depicting them by minimizing the variations as well as the orientation of any component with the overall flow of the circuit is important.
2. **Readability.** The aim of the schematic diagram is to convey to the reader a better understanding of how the circuit functions. This implies that the clustering of related components and the way to interconnect them should make it easy for the reader to comprehend.

Consistency of style and **good readability** are the two keys to a good schematic diagram!

DRAWING SCHEMATIC SYMBOLS

Visio comes with its own sets of engineering templates of shapes. The three related sets are the Basic Electrical, Circuits and Logic, and Systems.

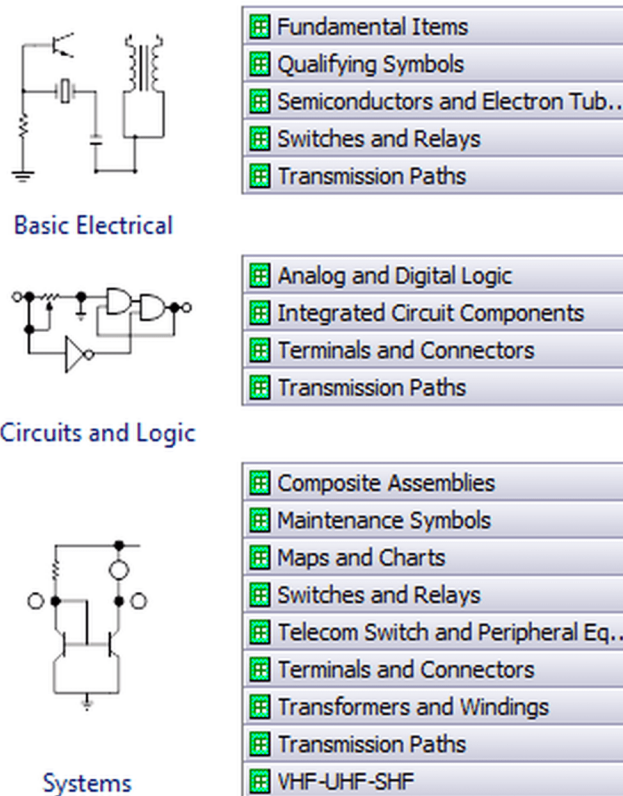


Figure 4.9 Visio templates under the engineering category.

Now don't get too excited or happy yet. Turn to [Appendices B-4](#) thru [B-7](#) and take a look at the available shape samples under some of these templates.

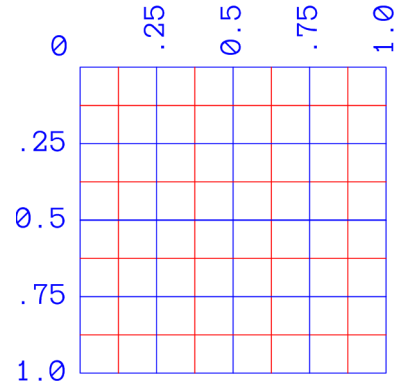
There's nothing really to shout about. Later versions of Microsoft Visio, depending on the licenses, may have more shapes but my gut feeling is that these shapes are construed by not-too-engineering minded people, which perhaps despite some artistic backgrounds, may not understand or have the right idea what engineering drawings are about when it comes to electronic illustrations and diagrams. This could well be the reason why there are companies out there providing better quality templates and shapes that suit different industrial requirements.¹³⁰

¹³⁰ One such company endorsed by Microsoft is Paul Herber's Sandrila Ltd. which offers templates ranging from electrical, electronics, to device packages. Their megapack suite has 535 stencils comprising 7,896 shapes and sells for \$72 which is quite a good bargain. You can go to their website at <http://www.sandrila.co.uk/visio-electrical/> to download some samples to try out.

Disappointment aside. In my opinion, it is better to create your own set of schematic symbols than to rely on the limited choices supplied by Visio. Of course, you can opt to buy online ready-drawn symbols, but for those with an engineer's eye for details, the styles and placements of component elements and label of these commercial packages may not be to your liking.¹³¹ Well, not to worry. In the pages that follow, I will show you how to draw these symbols—if you've read this far and like what you see in those schematic samples I put up, that is.

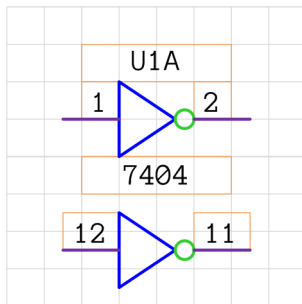
Base Reference Grid

It bears to repeat here that I use the imperial instead of the metric standard, so most if not all of my illustrations will be based on inches, specifically the industrial inch and not the decimal inch markings. So a one-square inch area with quarter (**blue**) and eighth (**red**) inch sub-divisions will look something like the figure on the right. Surprisingly, apart from large, multiple-pin components such as ICs, most discrete and logic symbols fit well within this one-inch area with room to spare. Therefore, I will base my illustrations on this grid reference which, if you follow closely, will ensure that your schematic symbols are well proportioned and laid out in the overall scheme of things.



Logic Gates

When we think of logic symbols, logic gates such as AND, OR, BUF, XOR, etc. usually come to mind. These are the basic elements of digital logics, including their inverted counterparts like NAND, NOR, NOT, XNOR, etc. Complicated logic symbols such as the tri-state gates, flip-flops, and multivibrators will require more effort to draw them.



Let's begin with a simple example—the NOT gate—specifically the 7404. It can be constructed using a triangle (**blue**) and a bubble circle (**green**). The input and output lines should be of sufficient length to accommodate both single- and double-digit pin numbers. Reference designation should be above and part number below the symbol. The text outlines (**orange**) are made visible for illustration purpose only and should not be present in the actual drawing. The text block margins formatting for the pin numbers is zero for left, right and bottom, and 2 points for top so the numbers stay close to the signal lines. The bubble is 1/16 inch diameter, in case you ask.

¹³¹ You would think that there should be some sort of standardization for the ANSI or IEC representation of components. Unfortunately there isn't. Symbols such as simple logic gates come in a number of variations, let alone the myriads of discrete components. There's even contentions as to whether the number zero should have a slash to differentiate it from the normal letter 'O', for that matter!

With this in mind, the basic logic gates can be represented thus:

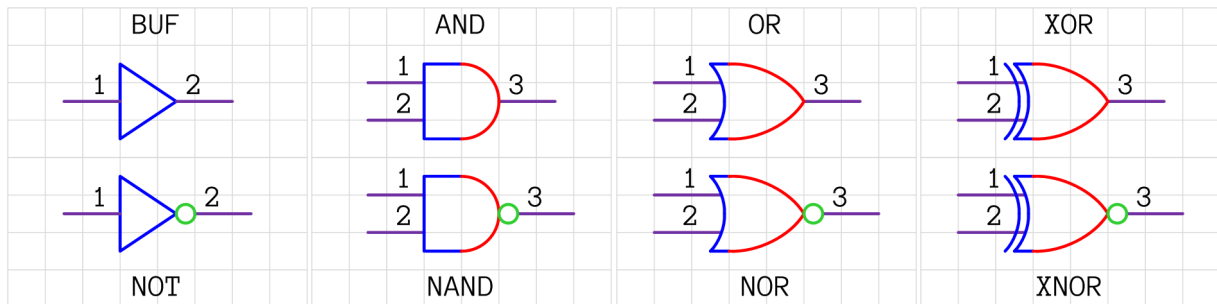
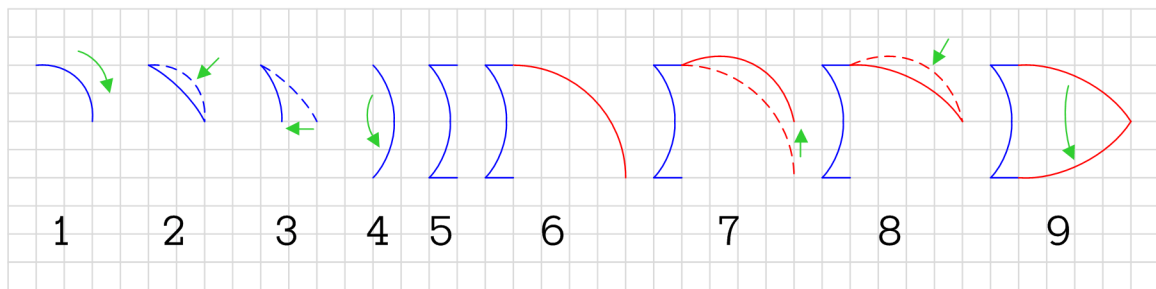


Figure 4.10 Basic logic gates, inverted and non-inverted.

The BUF, NOT, AND, and NAND gates are pretty straightforward to draw. The OR gate and its variants will need a little bit of tinkling, though. The following nine-step illustrations with accompanying notes should be self-explanatory:



Note:

1. The illustration has been scaled 250% of its original size for clarity.
2. Each square unit is 1/16 or 0.0625 square inch.
3. When re-shaping the arc¹³² in steps (2) and (8), zoom to 1400% for fine adjustments.
4. Steps (3) and (7) implies a move end point operation.
5. Steps (4) and (9) implies a duplicate shape followed by a vertical flip.

Interestingly, individual gates can be combined to form useful combinatorial logic circuits, like the R-S and J-K flip-flops. Indeed, some circuit designers make use of excess (or loose change) gates to create these kind of logic functions just to reduce IC counts while utilizing available resources to the maximum. So if you know your Boolean algebra well, it will not be difficult to figure out that a pair of NAND gates or a pair of NOR gates can be used to make an R-S flip-flop (see overleaf). This is one useful tip if you happen to work on a PCB with many logic gates.¹³³

¹³² Like all Visio shapes, an arc has its own handles when selected to allow manipulation—two end points for moving (stretching) and a mid-point for re-shaping. Just hover the mouse pointer over these handles when a shape is selected and a hint box will appear to tell you what you can do with that handle.

¹³³ I had my fill of the fun with a control board for a multiple outputs power supply unit.

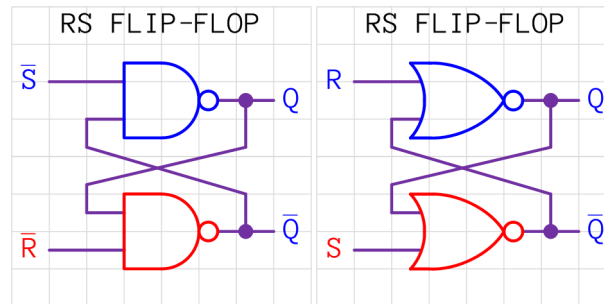
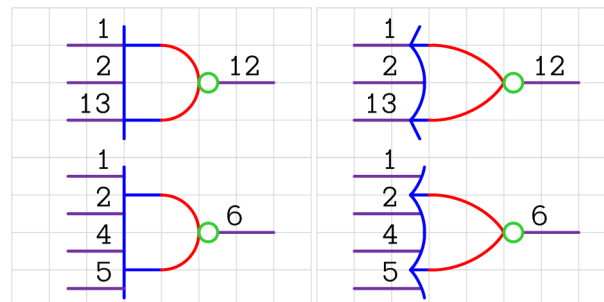


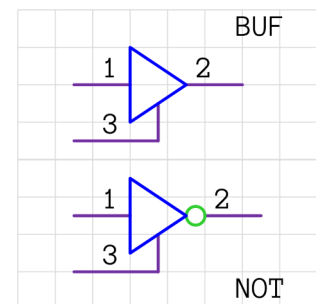
Figure 4.11 Seeing doubles (twins)—identical or fraternal?

What about gates with more than two inputs—such as the triple input NAND (7410) and NOR (7427), and the quad input NAND (7420) and NOR (7425)¹³⁴? How do you squeeze up to eight input lines within that limited space? The answer: simply extend the input boundary line!

Figure 4.12 Three's a crowd, and four's a group!¹³⁵

Tri-State Logic

The thing about digital logic is it's so predictably binary—1 (ON) or 0 (OFF)—that is, until you encounter a tri-state (or 3-state) gate that well, leaves you in an indeterminate state.¹³⁶ Thankfully, representing this on a drawing is easy enough for individual gates such as a BUF or a NOT. The usefulness of this kind of logic, however, are found in processor-based designs in which many peripheral components share common buses (address, data, etc.), thus requiring the use of multiple tri-state buffers or inverters controlled by a single ENABLE line, such as the 74244 and 74245.



We will learn how to draw such integrated logic ICs in another section, though.

¹³⁴ The 7425 dual 4-input NOR gates IC actually has an additional strobe line for each gate. I did not include it in the drawing in order not to distract my readers.

¹³⁵ And five's a party (alright!)

¹³⁶ This indeterminate state is also referred to as the high impedance state.

Flip-Flops

Like logic gates, flip-flops are an interesting lot to work with, and can sometimes appear as an individual entity or as a group in a circuit. The former are usually found in pairs in IC packages such as the popular 7474 and 74109:

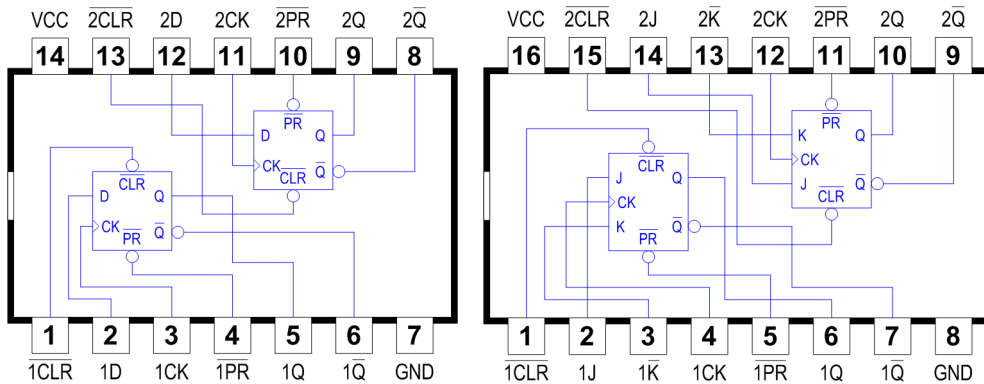


Figure 4.13 The D-type (7474) and J-K (74109) flip-flops.

When grouped, they usually come in fours (quad) or eights (octal):

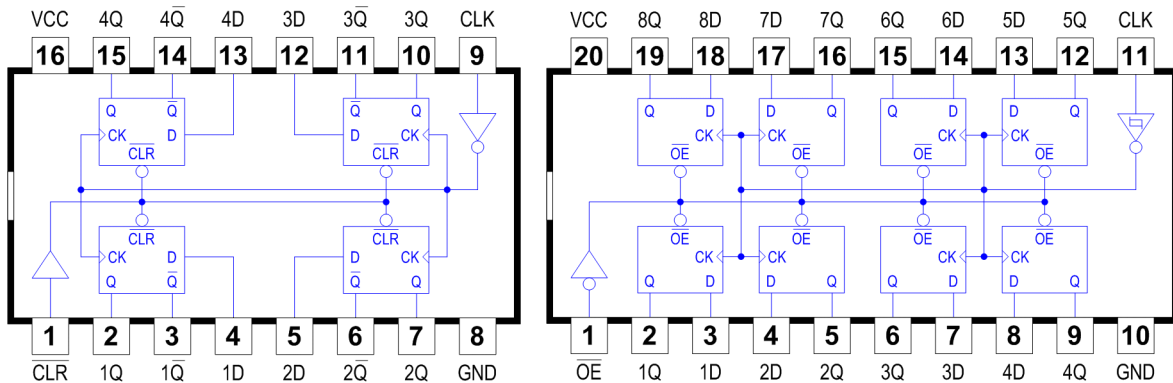


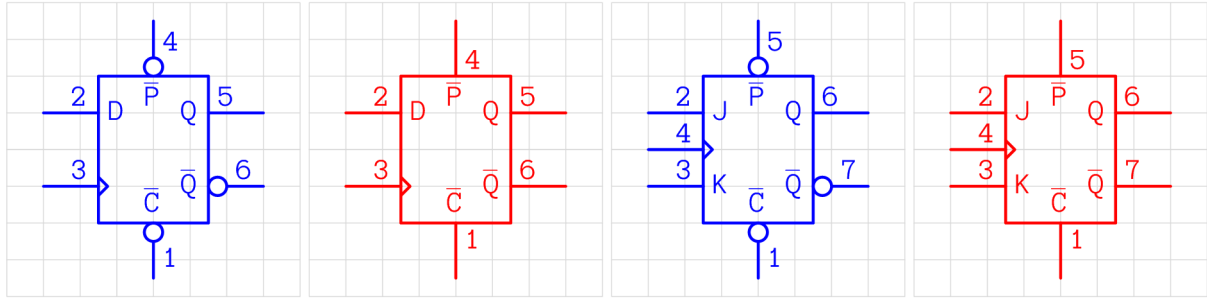
Figure 4.14 The D-type flip-flops in quad (74175) and octal (74374) arrays.

The above representations in IC packages are popularized by Texas Instruments 1985 The TTL Data Book and subsequently imitated by Motorola and other major digital IC manufacturers. I've included a list of illustrations for some of the common 54/74 TTL and 4000 series CMOS ICs in [Appendix D](#) pages [D-1](#) thru [D-4](#). They should come in handy when you need to refer to their pinouts as you encounter these ICs in your reverse engineering work.

Again, while individual flip-flops have their uses in discrete combinatorial logic circuits, the power of these digital elements are found in processor-based designs like their tri-state counterparts, latched or clocked synchronously. For the moment, we will focus on drawing the individual flip-flops and leave the integrated or grouped entities to a later section.

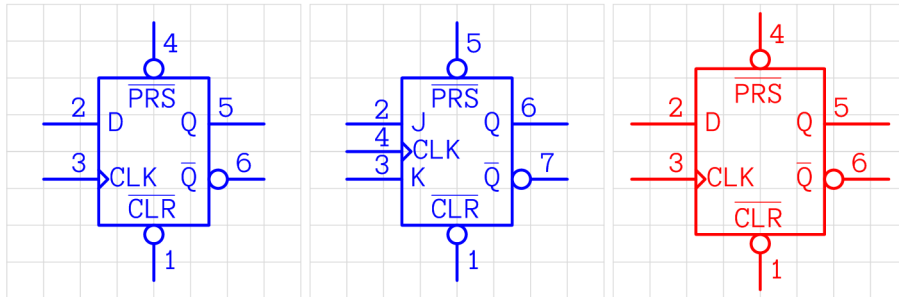
Chapter 4

We will look at the two most common types: the D-type and the J-K flip-flops. There are two ways to represent these logic entities, with the NOT bubble (blue) for active low signals (text with a bar on top), and without (red):



There is no right or wrong for not putting the NOT bubble in place, though with the NOT bubble it looks **clearer** as to the input and output signal definitions, but it certainly looks **neater** without. The choice is up to you but I'll need to remind you to be consistent, at least within the same schematic drawing.

Now, some of you may prefer to spell out the signal names in a more pronounced way like what you've seen in magazines and books, instead of simplifying. No problem with that too, except that adjustments are needed to accommodate the extra letters:



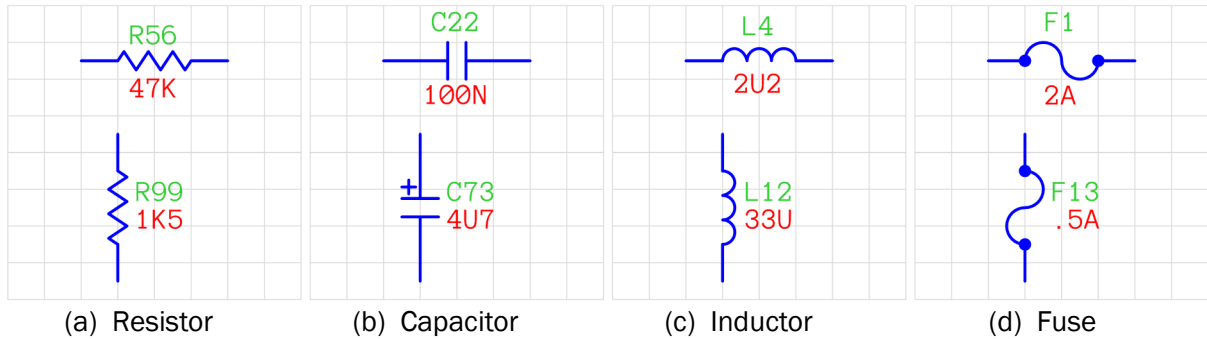
See how crammed these are compared to the simpler versions above? Of course, you can make the flip-flop's body taller and broader (red) but this will take up additional space, and in terms of aesthetics will probably look odd and out of balance with the surrounding logic gates. Insistence and consistence in styles are two conflicting issues that need to be addressed upfront—there are trade-offs to either—but the earlier you get it out of the way, the better you'll be able to focus on the actual work.

Before we take on the more complicated integrated circuits (ICs), let's look at some analog circuit elements for a change. You will find more tips and hints as to how to organize your schematics and look for possible design implementation in the PCB you're working on.

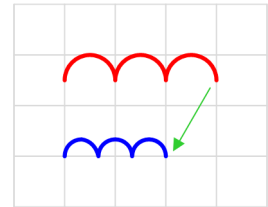
Here goes...

Passive Discretes

Passive components such as resistors, capacitors, inductors, fuses, transformers, etc. are common sight in analog PCBs, though the first three are also found abundantly in digital and mixed PCBs. Below are the schematic symbols in horizontal and vertical orientations. Note the placement of reference designations (green) and values (red):



The width of the resistor body is half a square (i.e. 1/16 or 0.0625 inch). The '+' sign for the vertical polarized capacitor is 3/4 of the 1/16 inch (you'll need to zoom in 1000% to have a better control on its placement). Fitting the inductor's three turns within two squares is done by first drawing three square turns and then shrinking it down to two from one corner handle. Lastly, the fuse is made up of two opposite semi-circles and joined by two filled dots at both ends.



Resistor networks can be drawn separately or bundled together, depending on how they are connected in the circuit. If the elements of a resistor network are connected to different parts of a circuit, you should identify them individually by their reference designations and values each, and if they share a common pull-up e.g. pin 1, then it's normal to label pin 1 for each occurrence of the elements of that resistor network.¹³⁷ If the resistor network happens to be tied to a bus in consecutive order, then it's more efficient to bundle them together and just use one set of reference designation and value to denote them together.

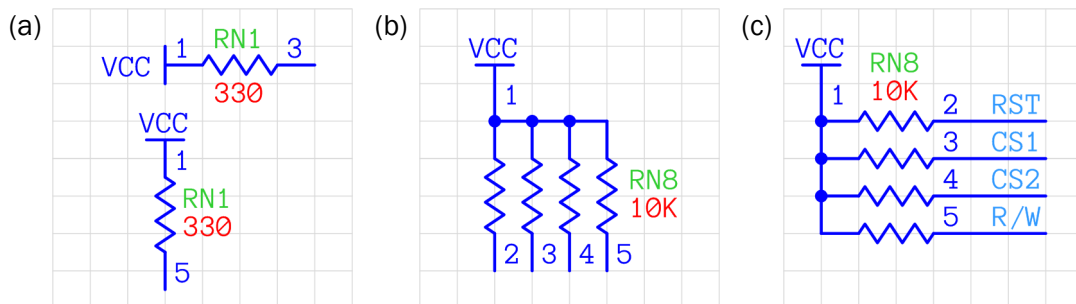
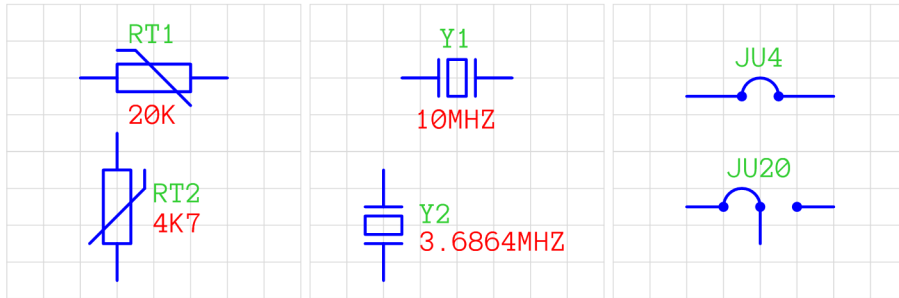


Figure 4.15 Resistor network: (a) individual element, (b) and (c) bundled elements.

¹³⁷ An alternative way of bundling individual elements is to use signal labels as connecting points. Refer to (c).

Thermistors, crystal oscillators, jumpers, and other passive discretes can also be depicted in the same manner as shown below:



Power and Ground

Sometimes there may be more than one voltage source or ground type in a PCB. Unless you're sure of the voltage values, it'll be better to use standard voltage designations like VCC, VEE and VSS to denote them and not +5V, +15V and -15V.¹³⁸

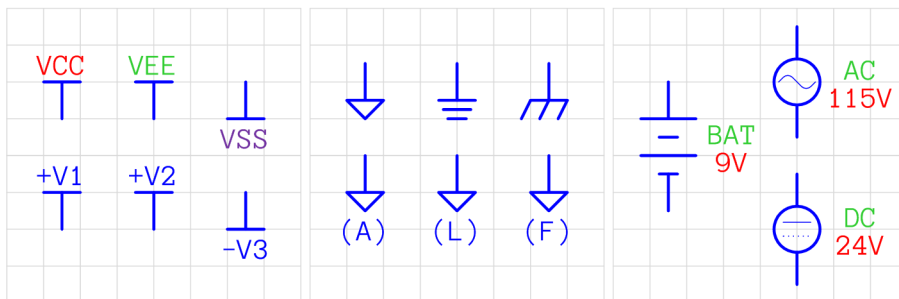
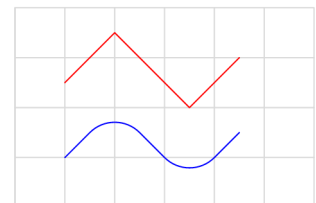


Figure 4.16 Types of power and ground symbols.

Similarly, there may be various types of ground references in a PCB, so it is important to denote them correctly. Figure 4.16 top row ground symbols are: signal, chassis and earth groundings, respectively. It may be useful, even necessary, to denote the type of signal ground by placing a letter within bracket markers, like (A) for analog ground, (L) for logic ground, and (F) for floating ground, etc.

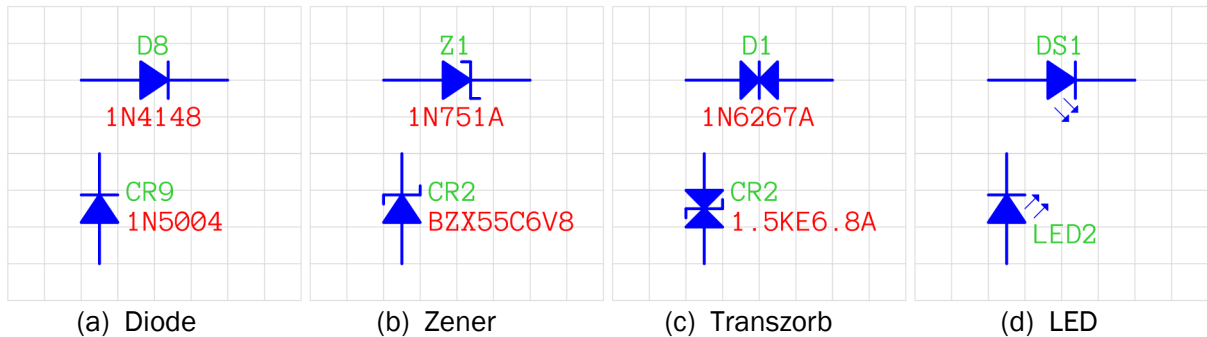
Another type of power source symbols are the battery and generators. While the battery may be found in some PCBs which use lithium as a CMOS back-up source, the AC and DC generators are seldom, if ever, required unless you need to include them as external sources in your schematic diagrams. Note that the sinusoidal waveform is created using joined straight lines (red) and changing the corner rounding value to 0.125 to achieve the effect (blue).



¹³⁸ Note that VCC does not necessary mean +5V, and VEE and VSS do not translate to +15V and -15V automatically. Some logic operates at a higher voltage than standard TTLs, and some dual rail voltages may go as high as $\pm 30V$ for that matter! See also footnote 127.

Semiconductors

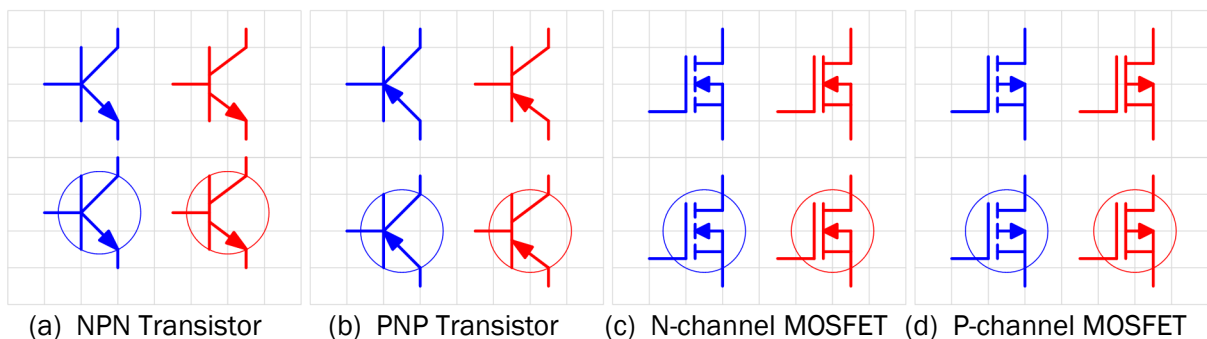
Two-terminal semiconductors such as diodes, zeners, transzorbis and LEDs can pretty much be drawn like passive discretes:



Three things to note here:

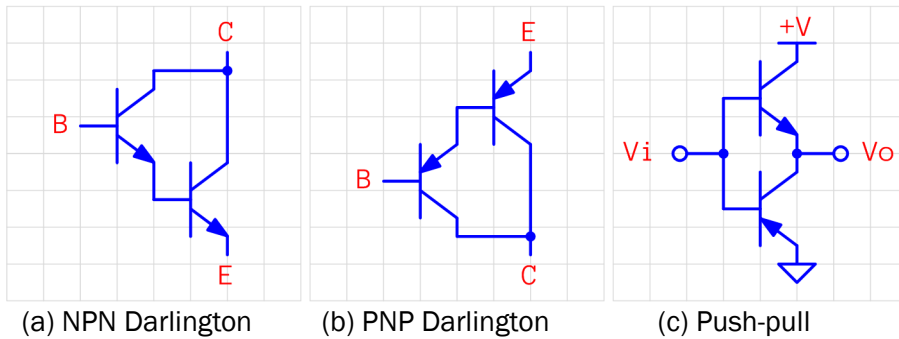
- The body of the diode is 3/4 of the 1/16-inch square and can be aligned center or middle to one single line, unlike two for the resistor, capacitor or inductor, since the solid-filled body of the diode covers over the line.
- The part numbers can be quite long for some zeners and transzorbis, so you may want to just indicate the breakdown voltages instead of their full part numbers.
- Light-emitting diodes (LEDs) do not normally have part numbers printed on them, so you may want to leave them out to make room for the arrows.

Three-terminal semiconductors such as transistors and MOSFETs have two configurations—NPN and PNP, so:



Two common ways of depicting transistors and MOSFETs are with and without circle enclosures, each having different appeal to different groups of people. For transistors, there is even a distinction in terms of the junctions being joined (**blue**) or separated (**red**). For MOSFETs, it's the depletion nodes—separated (**blue**) for enhancement node, and joined (**red**) for depletion node.

Transistors can be paired up to form some useful circuits, such as the Darlington pair and the push-pull amplifier:

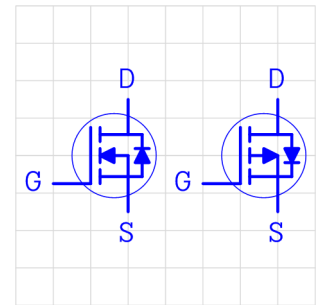


MOSFETs are often used in high voltage or power switching applications, and are subject to possible failure due to excessive maximum drain-source voltage (V_{DS}).¹³⁹ For this reason, most MOSFETs come with protection or TVS diodes across their drain-source terminals. Depending on the type of circuit MOSFETs operate in, it may even be necessary to incorporate a TVS diode at the gate to source input for added protection from ESD or input transients.

A Little History

There's a little story in the footnote below which I think would make for an interesting read. I still remember it clearly just like it happened only yesterday.

And yes—in case you're wondering—we still have those remaining BUT34s and MTM15N5 with us till this day...



¹³⁹ Back in mid 1990 when I first joined the company, there was an engineer who worked on designing an active load for a 2KV power supply output. He needed to load the 2KV output until it cuts off at around 1.65A. It was still the MS-DOS era when Windows 3.0 just made its debut, and there was not many analog simulators available on the PC platform.

I saw him wiring up a crude prototype using metal-clad resistors in series on terminals mounted atop an aluminum sheet and using a BUT34 power TO-3 transistor as his active load, biased by a manual control circuit at its base. While I watched, the BUT34 simply blew up when the current reached around 1.35A and he would replace a new transistor, then tweaked the values of the metal-clad resistors and try again. Right beside him was a box of about ten BUT34s, waiting to be sacrificed. After a few failed attempts, I told him it's probably not going to work.

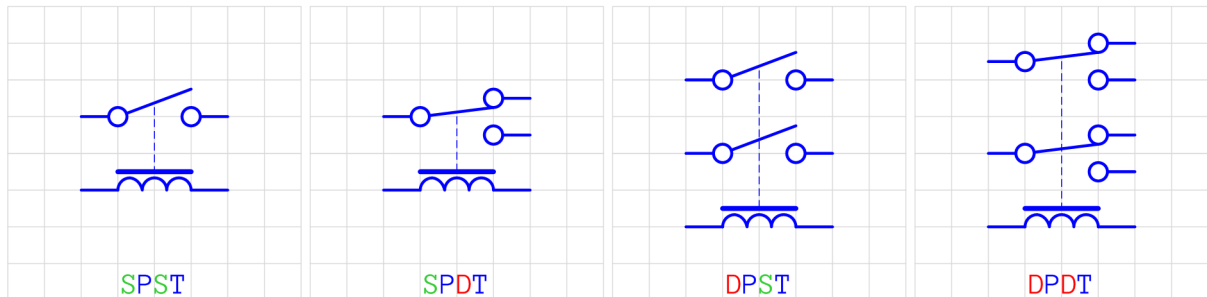
Fortunately, there was a student version of Microcap II in one of the new NEC 286 PCs in the workshop, so I suggested simulating his circuit using the parameters of the BUT34. It was spot on! When the current value reached about 1.35A, the transient analysis graph suddenly shot up and hit the roof. "There's the problem," I told him, "the safe operating region is not sufficient to handle the current requirement of the 2KV load."

We then proceeded to source for an alternative by scavenging through the data books in the library, and decided on a MOSFET which seemed to fit the bill. It was an MTM15N5. We did another simulation run and this time at slightly over 1.65A the current shutdown to zero instead of shoot up. The engineer placed ordered for five pieces and when they finally arrived, he quickly fitted it in the load circuit and gave it a go. And you know what? It worked on the first try! Naturally, he was grateful to me for saving the day.

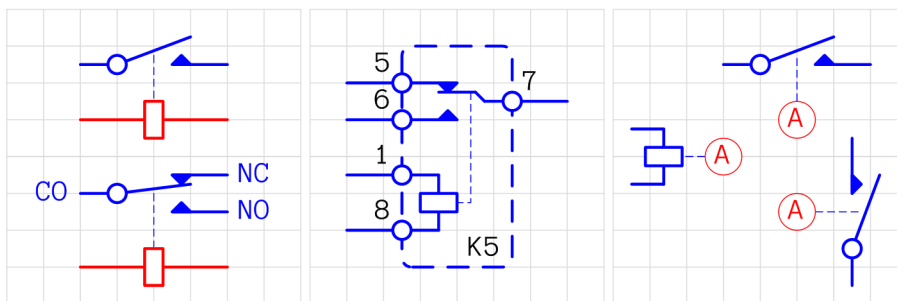
Relays

There are two types of relays, as mentioned in Chapter 2—electromechanical and solid-state (SSR). The former has make-and-break contacts that are energized or de-energized by a relay coil via a control voltage, and is also known as an electrically operated switch; the latter uses semiconductor switching without moving parts and can attain faster speed, but has limited switch configurations, and is also known as an analog switch. We will look at drawing the electromechanical type here, and touch on the solid-state type when we discuss ICs.

Here are the four basic configurations:



On a realistic level, the schematic symbols are usually drawn with the normally open (NO) and normally closed (NC) contacts represented as wedge shapes while the common (CO) is the usual bubble. You can also simplify the relay coil (red) further. For example, the simplified SPST and SPDT on the left, an actual implementation (K5) on one of the PCBs I worked on in the middle, and in the case of a DPST that are separated because they are used in different parts of a circuit, labeled bubbles (A) can be used to link them up as shown on the right:



There are no hard and fast rules when it comes to drawing relays, especially electromechanical ones for all the basic configurations; only two guidelines need to be observed:

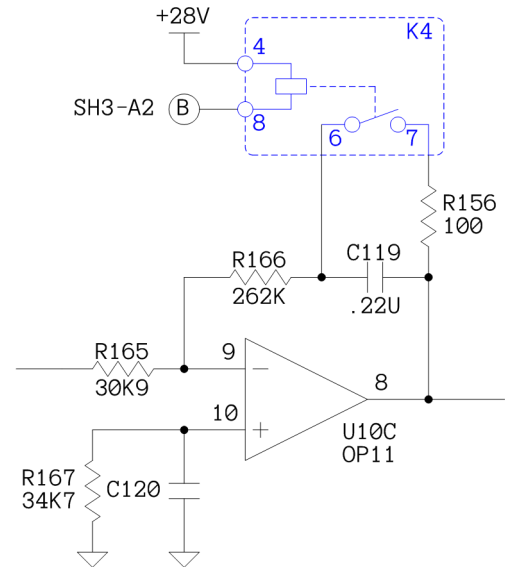
1. Symbols must be proportionally scaled with respect to the rest of the circuit, for obvious aesthetic reasons, and
2. They should not obscure the flow or interpretation of the circuit in which they are found.

Take the sample circuit to the right for example:

It is not uncommon to find relays being used in feedback loops of operational amplifier (op amp) circuits, whether to switch in or shunt out components to effect a change in its gain or alter its integrating characteristics.

You should therefore take note when working on analog PCBs with lots of op amp ICs and relays, electromechanical or solid-state alike.

Talking about op amps, that's the next thing on our menu.

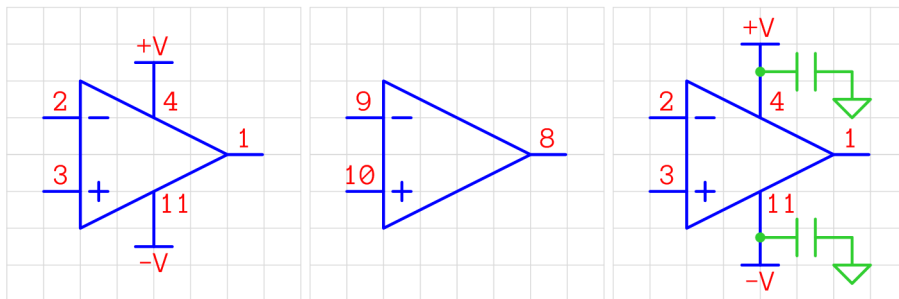


Op Amps

Operational amplifiers, or op amps in short, are generally found in analog circuits that require signal conditioning, computation, transformation, etc. due to their efficiency and versatility. With proper selection of feedback elements, op amps can perform addition, subtraction, averaging, integrating, and differentiating functions. Some basic op amp configurations can be found in [Appendix D-5](#).

As feedback techniques become established and more widely known, op amp circuits begin to find greater usage in modern control and instrumentation designs, especially in AC and DC amplifiers, high precision oscillators, comparators, servo motor control, AC to DC converters, deflection yoke drivers, etc. In fact, the kind of op amp applications that the designer can come up with are limited only by their knowledge of the device itself and their own ingenuity to invent and innovate.

Single op amp IC are usually drawn together with its rail voltages (left), and sometimes with decoupling capacitors (right) if space permit. For dual or quad op amp ICs, only one element¹⁴⁰ needs to carry the rail voltages symbols; the other remaining op amp(s) do not have to (middle).



¹⁴⁰ Usually it's the first element but you can choose any one of the elements that preferably will not obscure or affect your schematic's layout.

Take note that the inverting (-) and non-inverting (+) symbols of the op amp are lines and not text. You may opt to use the '+' and '-' but the effect is not as good because the '-' tend to be narrower than the '+', which creates a sense of imbalance to the overall op amp symbol.

Consider the following oscillator circuit which makes use of a combination of op amps (U1) and voltage comparators (U2) to generate four phase-shifted triangular waveforms with the same amplitude and frequency:

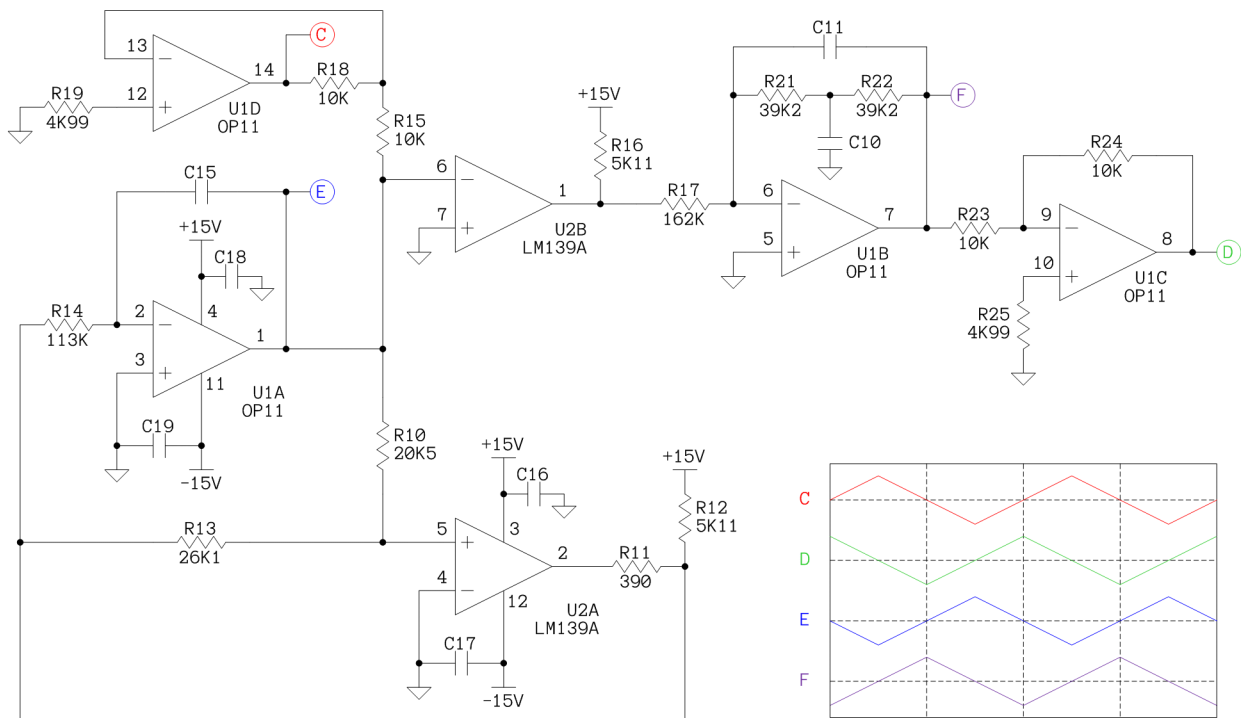


Figure 4.17 Op amp oscillator circuit for PWM reference application.

Notice that the first elements of U1 and U2 have the rail voltages appended while the remaining elements do not. For the negative rails of both ICs, the same ground symbol is also shared to reduce clutter. This brings out another point worthy of note—where possible, symbols should be reused within close proximity of entities that share them, such as power and ground.

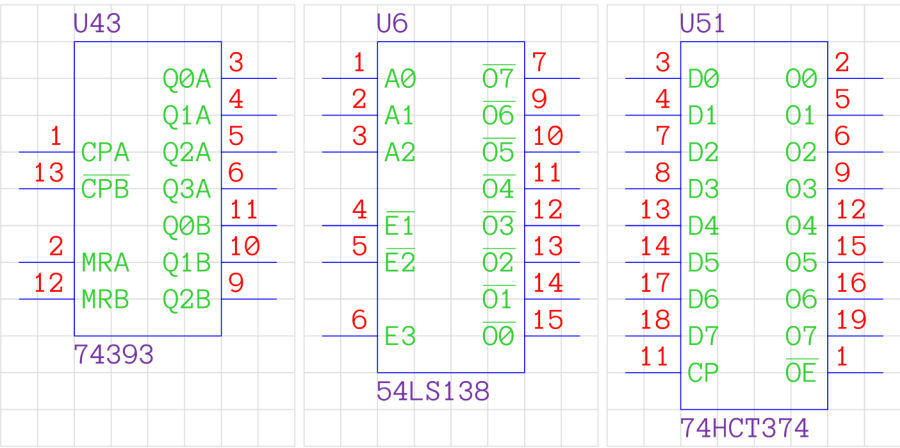
Referencing [Appendix D-5](#), can you determine what kind of configurations the elements of U1 are used? (For answers, see footnote below.¹⁴¹)

¹⁴¹ U1A – Integrator, U1B – Integrator with T-filter, U1C and U1D – Inverting amplifiers.

ICs

Though the op amp belongs to the IC category, it was given a section of its own because of its peculiar representation in analog circuits. The same can be said of logic gates which were also given a separate treatment in the beginning, by way of representation in digital designs. I've decided to group the rest of the ICs, digital or analog, in this section because the way of representing them symbolically is quite similar, and because of pin counts and functionality that require them to be drawn as a whole, cannot be simply confined within the one-square inch grid that we've used so far.

Consider the following three TTL ICs with 14, 16 and 20 pins:¹⁴²



U43 is a dual 4-bit binary counter (74393), U6 a 3-to-8 line decoder (54LS38), and U51 an octal D-type flip-flops (74HCT374). Notice that the VCC and GND pins for these ICs are not shown in the symbols. As a rule of thumb, you should not include these power and ground pins for TTL and CMOS ICs, or any digital ICs that have similar power and ground pin arrangements like the TTL/CMOS ICs.¹⁴³

The shapes are straightforward to draw, with only one point to bear in mind—signals should flow from left to right as far as possible (see [Circuit Layout and Signal Flow](#) on page 133). The text block formatting values are as follows:

	Top	Bottom	Left	Right	Align
Reference designator (purple)	1	0	0	0	Left
Part number (purple)	1	0	0	0	Left
Left pin numbers (red)	2	0	0	2	Right
Right pin numbers (red)	2	0	2	0	Left
Left signal labels (green)	1	0	2	0	Left
Right signal labels (green)	1	0	0	2	Right

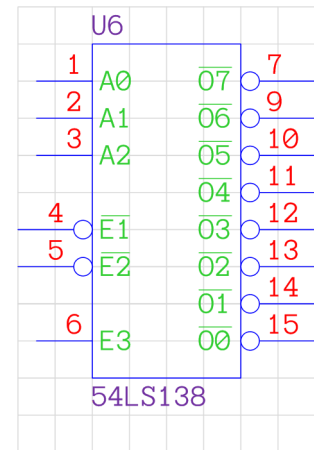
¹⁴² Surprisingly, I can't recall seeing any 18-pin TTL IC, or for that matter, any 18-pin CMOS IC either!

¹⁴³ The reasons and logic are explained in [Elements of a Good Schematic Diagram](#) under [Other Considerations](#) (pages 135-136).

These are suggested values only so change them to your liking as you see fit. Signal labels with macrons (an overhead $\bar{}$) such as CPB or E1 are created using lines, since Visio does not have equation editing feature like Microsoft Word. Fine placement of bars or lines over the signal labels are achieved by zooming in and using the arrow keys to move the lines in incremental steps while they are selected.

You can add bubbles to input and output pins that are active low i.e. signal labels with macrons to emphasize the logic characteristics of the pins. For example, U6 (54LS138) above can be re-drawn as shown on the right. Note however, that the pin numbers will be offset by the width of the bubble as a result. In some of my past schematics I did include these bubbles for better clarity but soon learnt that it has its own set of problems:

- It takes up valuable space in my drawing, especially if there are many ICs that have signal labels with macrons.
- You need to make extra effort to remember to bubble these signal labels to ensure consistency.
- Datasheets of the same IC may have different naming conventions you might miss out on some signal labels and find yourself without that extra space needed to include the bubbles later on.



If you've listened to enough of my ranting about why using bubbles on IC symbols other than logic gates is not only a bad idea but an added burden, you would definitely be better off without having to bother about them. Just limit its use to pure logic gates and you'll be glad you heed my advice!

The next thing I want to talk about is the signal label. You'd notice that for standard ICs up to 20 pins which I've drawn so far, their widths are fixed at one-half or 0.5 inch (four small squares). In fact, you can use this width for ICs with pin counts of up to 24 (or even 28!).¹⁴⁴ The trick is to simplify the signal labels indicated on the datasheets of the ICs to no more than three to four letters without losing their meaning, if possible.¹⁴⁵

There is no stopping you from using a wider option though, especially if you have difficulty fitting signal labels that are longer than five letters within the default width that I suggested. This is true with ADCs and DACs which tend to have rather elaborate pin names that can be quite a challenge to shorten or simplify—it takes some thinking but is definitely still do-able.

Let's look at one example, an AD574A. This is a complete 12-bit A/D converter which has a 28-pin count. For ICs of such nature, I usually widen it to 5/8 or 0.625 inch and try to simplify the signal labels to no more than six letters. Turn overleaf for the solution.

¹⁴⁴ Some memory ICs such as dynamic RAMs (DRAMs) can go up to 32 pins and I still maintained their width at one-half inch to accommodate as many of them within a drawing page as I could. The only time I'd split them was if they belong to different arrays or groups.

¹⁴⁵ Refer to page 134 under [Text and Labels](#) for examples.

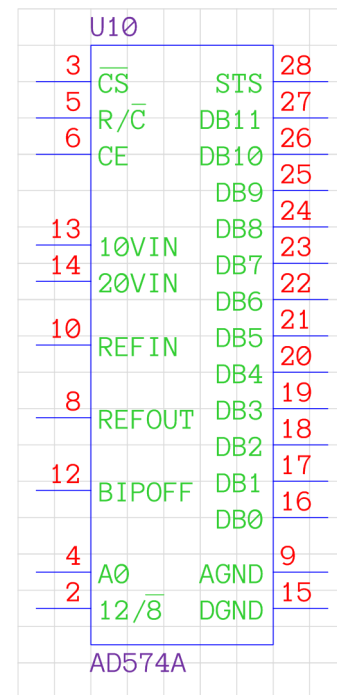
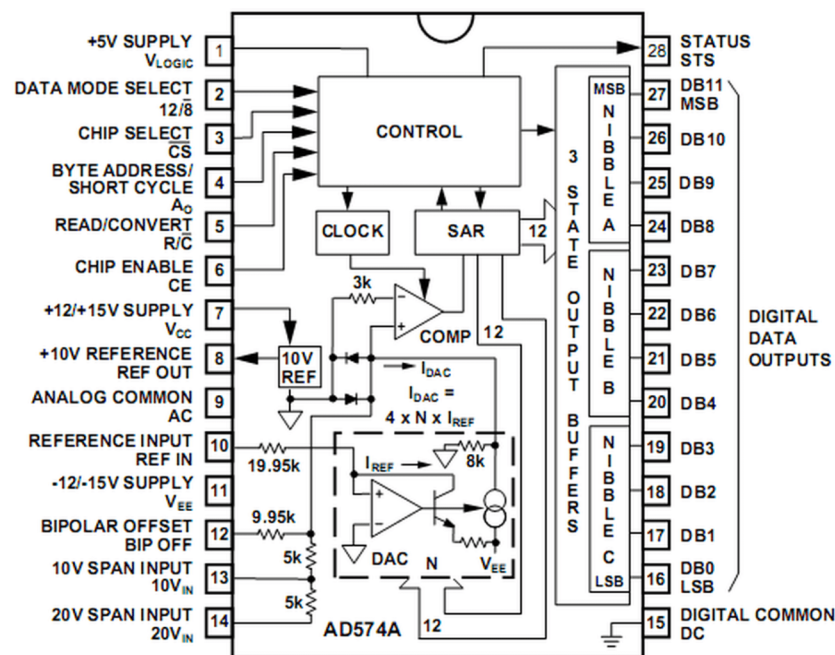


Figure 4.18 AD574A datasheet pin-out and symbol representation.

If you try to tally the number of pins for both diagrams, the IC symbol on the right will come out short of three, which are the +5V Supply or V_{LOGIC} (1), +12/+15V Supply or V_{CC} (7), and -12/-15V Supply or V_{EE} (11). These power pins are assigned to their respective decoupling capacitors on the first page of my drawing with the rest of the decoupling capacitors to group them within one sheet for ease of reference. Additionally, I've indicated in the space above it the missing power pins and their associated voltages (see Figure 4.19). This is another useful way of information presentation in schematics.

Now take a look at how I reduce and simplify those lengthy signal labels on the datasheet pin-out to the schematic symbol:

Data Mode Select 12/8	12/8	Reference Output REFOUT	REFOUT
Chip Select $\overline{CS}$	$\overline{CS}$	Reference Input REFIN	REFIN
Byte Address/Short Cycle A_0	A_0	Bipolar Offset BIP OFF	BIPOFF
Read/Convert $R/\overline{C}$	$R/\overline{C}$	10V Span Input 10VIN	10VIN
Chip Enable CE	CE	Status STS	STS
Analog Common AC	AGND	Digital Common DC	DGND

I have opted not to retain the short form for the Analog and Digital Common because the AC and DC can be a misnomer compared to the more well-known AGND and DGND. Also, take note how I grouped the signal pins so that they matched with the signal lines from adjacent components. It does take a bit of practice to get it right eventually, but you'll get there.

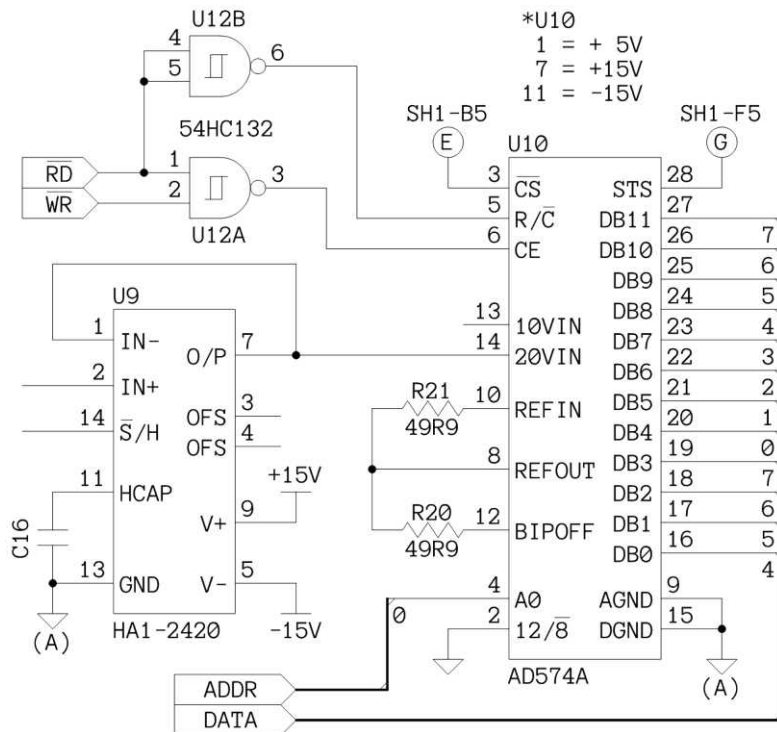
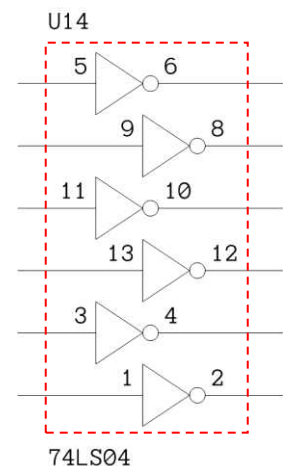


Figure 4.19 P/O schematic diagram showing the AD574A.

There's quite a bit of things we can say about the above partial diagram, such as the buses and the signal labels, but I'll leave them to the next two sections.

I want to draw your attention to the two Schmitt trigger type two-input NAND gates (U12) on the top left. Notice that I only used one part number for the two logic gates right in their middle, and chose to place the reference designator for the lower gate (U12A) at the bottom instead of the top.

When two or more similar logic gates of the same reference designation are in close proximity to each other, it's good to reduce clutter by removing any extra part numbers since one is enough to identify them all. In fact, if all the elements of the IC are present in one place, like the hex inverters (U14) on the right, you might want to group them together with a dotted box and use only a single set of reference designator and part number for them, and not have to bother with which is U14A, U14B, U14C etc.



Efficient use of space is also a plus point if you want to squeeze as much of the related circuitry into one sheet as possible. If you're short of height (pun intended) and have the width to spare (again), you can stagger the logic gates like U14; if it's the other way around, then simply line the logic gates in a straight column.

Before I forget, let's look at an example of a solid-state relay (SSR):

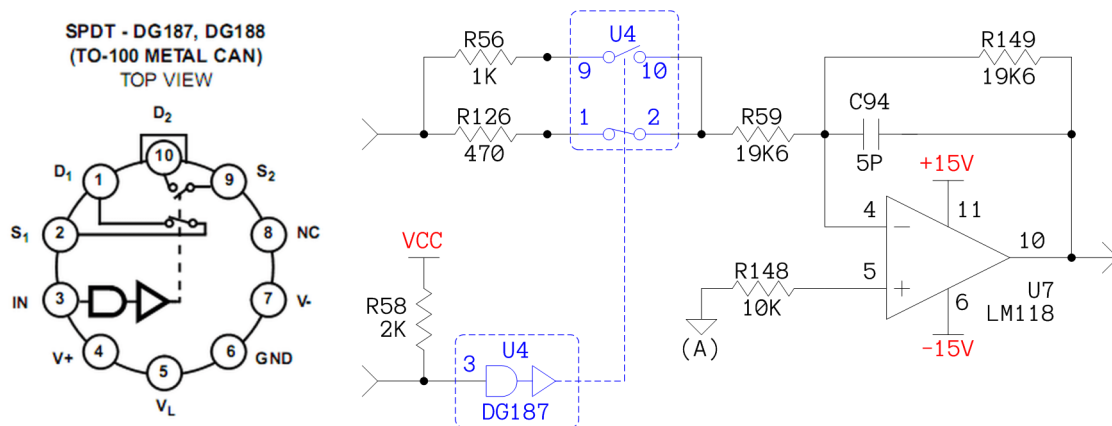
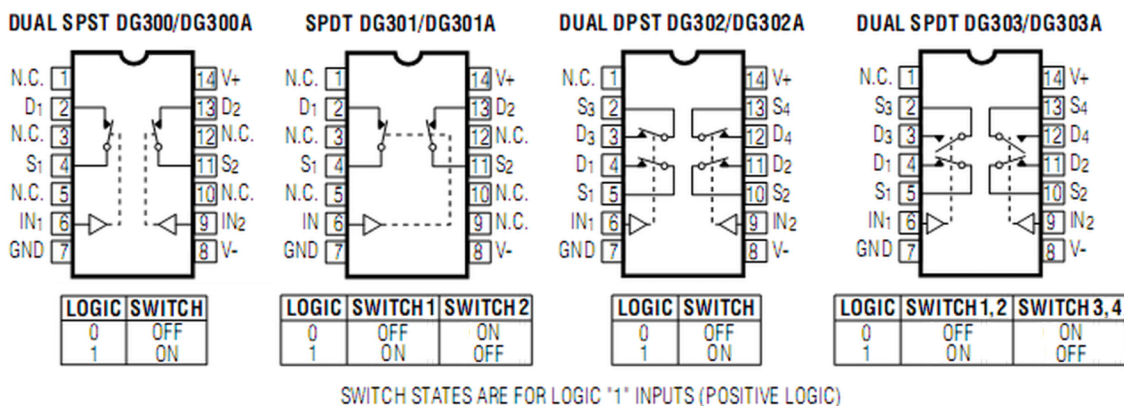


Figure 4.20 DG187 pin-out and op amp circuit application.

Just like its electromechanical cousin, the DG187 can be used in op amp circuits to effect gain changes as shown in Figure 4.20 above, with its pair of SPDT (NO-NC) solid-state contacts. The way of representing is similar except that the energizing medium is a logic gate driver instead of relay coil.

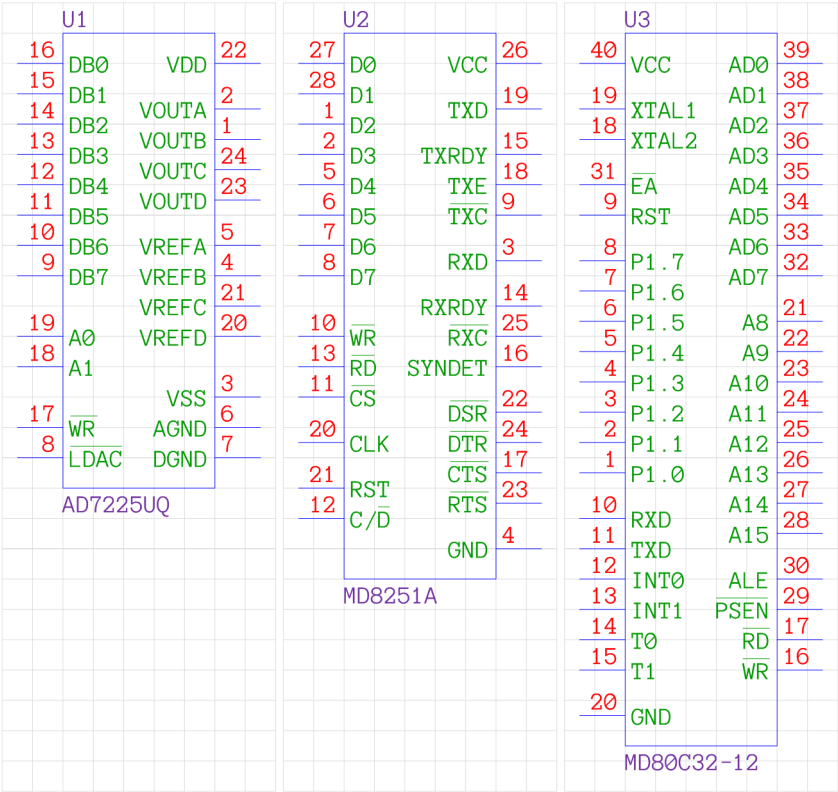
Unlike the electromechanical relay which has only one relay coil per module, solid-state relays can have between one to four driver-contact pairs because of their integrated circuit constructions. This offers significant space savings over the electromechanical type, especially when a PCB requires the use of many relays for programmability and configurability of circuit topologies involving multiple op amps.

Below are some DIP package solid-state relay ICs of the DG300-series:

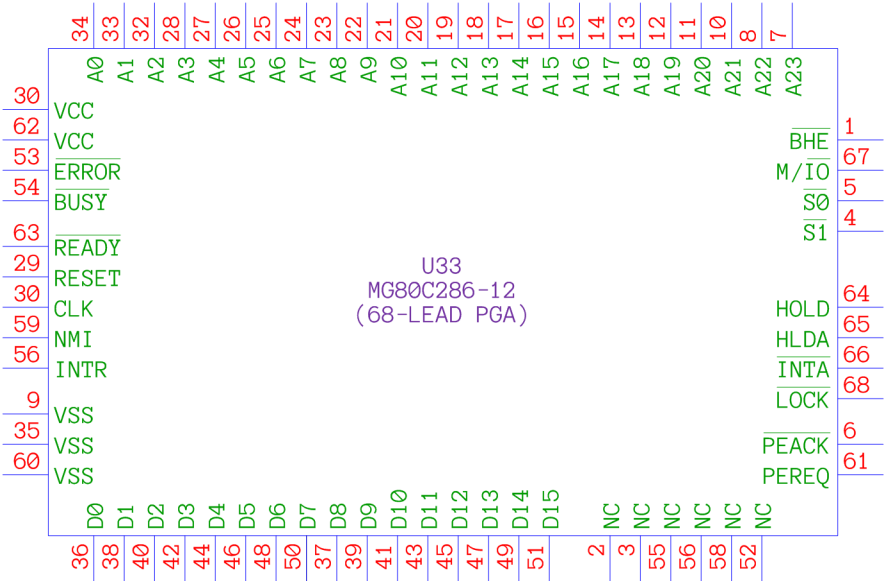


One thing to take note is that unlike electromechanical relays, opened contacts in solid-state relays have lower isolation resistance and closed contacts exhibit a small junction impedance, instead of open and short for mechanical contacts. Therefore, it's important to be aware which end of the SSR's contact you're taking reference when doing connectivity tracing so you don't get misled.

Here are a few more large-scale ICs to give you a sense of scale and proportion:



IC symbols are not limited to vertical rectangular forms. Many PGA, PLCC package processors and FPGAs with pin counts exceeding the 68-pin limit of DIP package ICs are often represented like the 80C286 IC in the following figure:



Buses

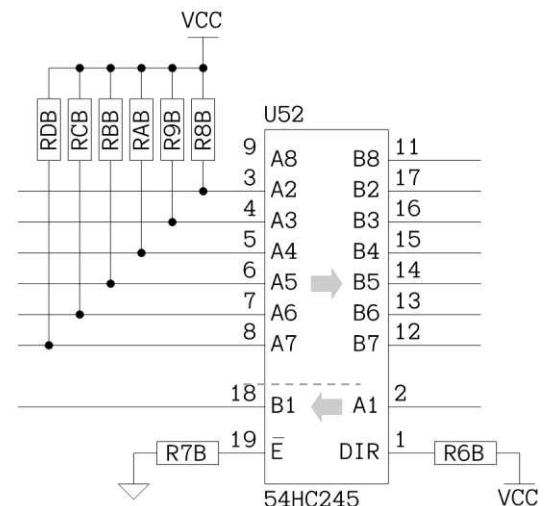
Buses are a useful and effective way to organize big groups of related signals, such as addresses, data, and even control lines. There are different ways of representing buses in how they converge and fan out their individual signal lines from the source to the destinations—device to device, device to connector, connector to device, connector to connector, etc.

Figure 4.19 provides a glimpse of how a 12-bit data bus can be wired from another part of a circuit with a connecting signal label, but it is not sufficient to illustrate the power of using buses in a PCB that contains multiple networks of signal groups. We will look at a more elaborate example in Figure 4.22 on [pages 158-159](#) instead. Buses are such important elements in digital PCBs, specifically CPU-based PCBs, that there is much to discuss about their usage.

Starting with the connector P1 from the left, you can see that address lines A1-A16 connect directly to U53 and U54 while data lines D0-D15 are wired to U56 and U56, respectively, all of which are octal bus transceivers (54HC245). The '_2' after the signal names is used to denote which group the address and data lines belong to—in this case group two, just as there are '_5' for group five and '_CPU' for those related to the CPU itself. This is one quick and easy way to organize multiple address and data groups. Notice also that there are arrow symbols associated with each 54HC245 IC to indicate unidirectional for address lines and bidirectional for data lines, as evidence by how the enable (E) and direction (DIR) pins are configured or controlled.¹⁴⁶

In the middle of the connector is a group of input signals that are pulled up to ensure they settle at expected logic levels if they are disconnected, or if high-impedance when they are disabled. Because of these pull-up resistors,¹⁴⁷ a bus is used which bends like a U-shape around them. Incidentally, a bus' width is double that of a signal line to differentiate the two.

Of course, there is nothing wrong if you decide to use the schematic representation on the right instead of a bus—after all, it's not going anywhere else. It's all a matter of personal preference and style, and making the best use of space available.



The buffered address and data lines then route out to different groups of memory devices (RAMs and EPROM), either directly or through another one or two level of transceivers. And here's where the buses' number references and their direction indicators come into play. Consider the following portion of the schematic:

¹⁴⁶ Normally there is no need to include this extra arrow feature in your schematic symbol, but in this instance due to the number of transceiver ICs (54HC245) present, it can be quite confusing navigating around the different levels of interconnection without these indicators as visual aids to ascertain correctness. If you find it odd, you can remove them after you've completed and verified your schematic diagram.

¹⁴⁷ The resistors used in this schematic diagram are not the usual symbol with their reference designations and values indicated, but simple rectangles with reference designations embedded. The reason is because these are individual SMD resistors of mostly similar values for pull-ups (4.64K) and pull-downs (330) and this representation saves space and requires less effort to keep tabs while working on the schematic layout.

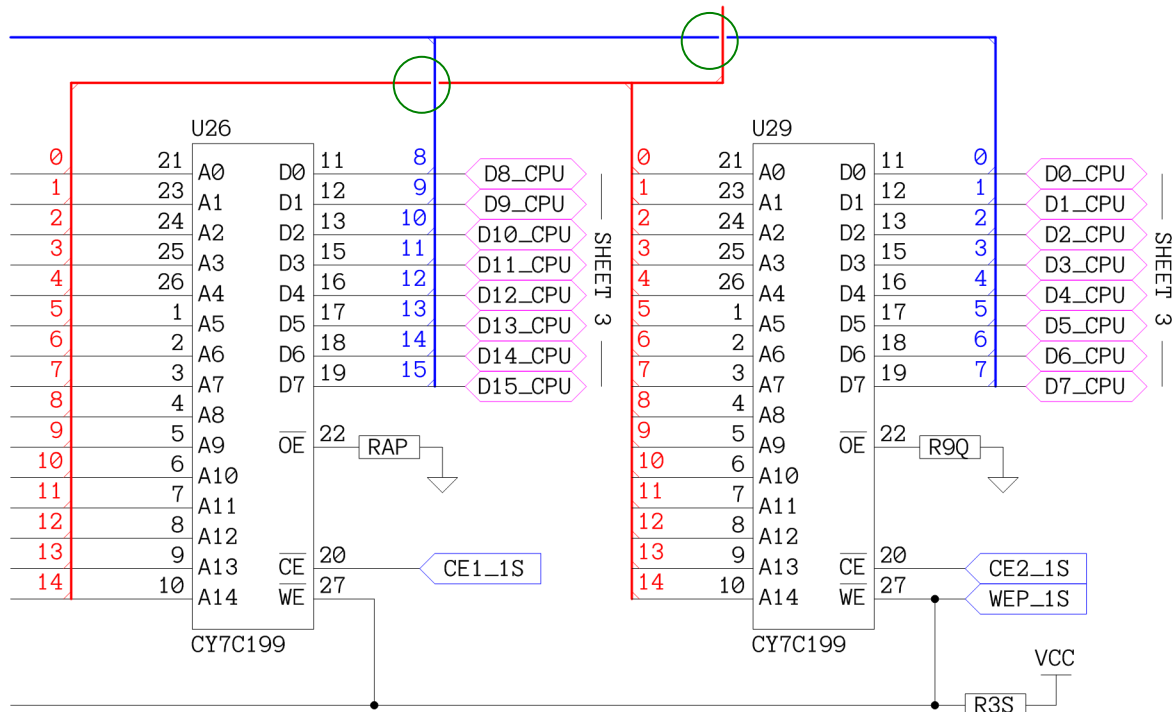


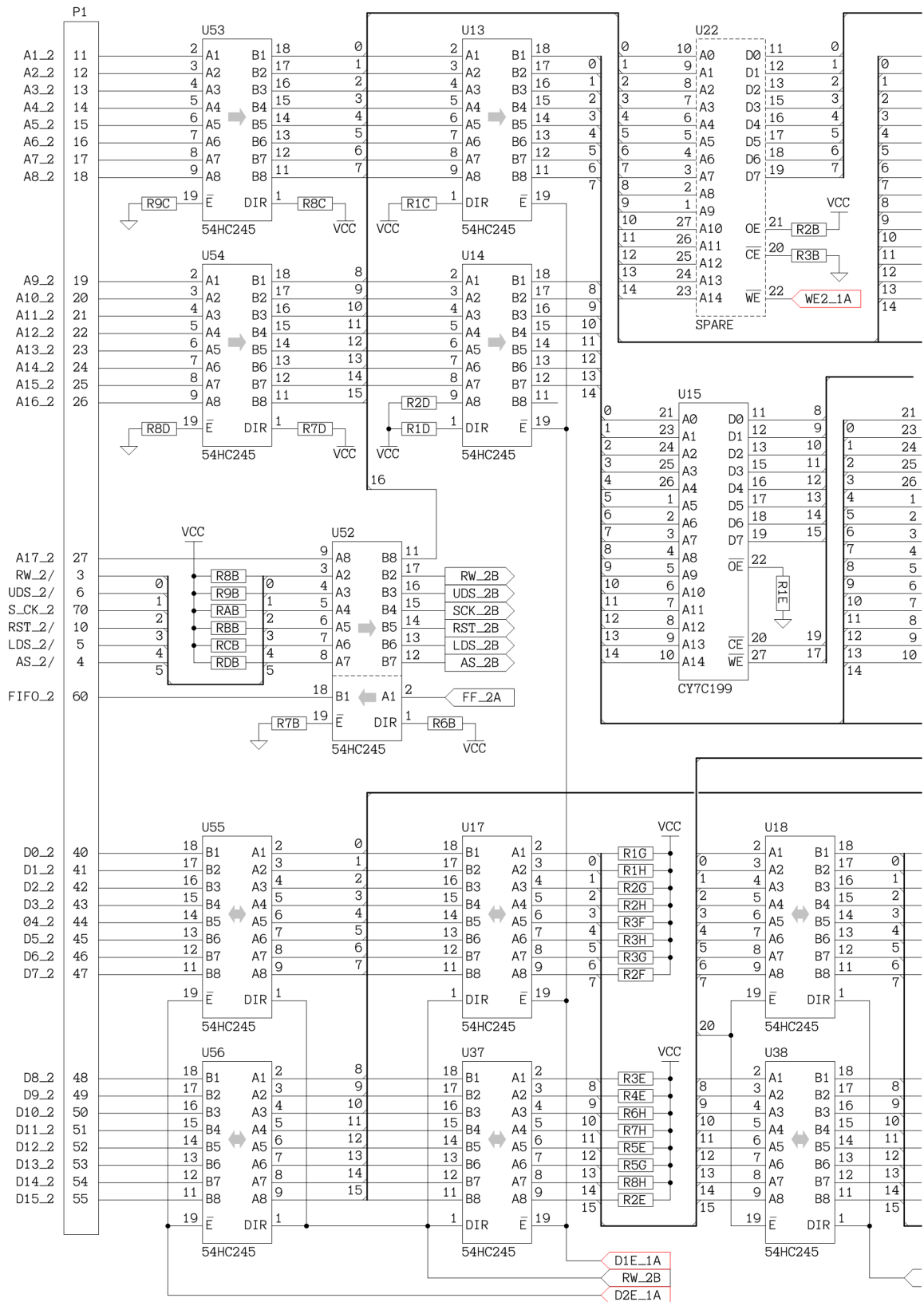
Figure 4.21 Address and data buses for static RAMs.

The address lines (red) have number references 0-14 and the data lines (blue) have number references 0-7 (low byte) and 8-15 (high byte). Sometimes additional lines not belonging to data lines are found tagged to the data bus as well; these are control lines and it's one way to reduce clutter or to circumvent lack of space by hitch-hiking related signals on the same bus that goes to the ICs of interest.

Direction indicators are those tiny 45-degree line marks beside the number references and at the converging corners of buses. These markers provide the sense of flow of the bus signals from the origin to the destination as well as in-betweens.

Another thing to take note are the breaks on horizontal buses as the vertical buses cross them, such as that shown in the green circle regions above. The break is achieved by overlaying a white filled rectangle over the horizontal bus and under the vertical bus. While it may not be necessary to break the bus lines, it does help to segregate them especially if they are not colored.

If possible, align ICs that have similar connectivity styles, such as the address transceiver pairs of U53, U54 and U13, U14 with respect to the data transceiver pairs of U55, U56 and U17, U37. Study Figure 4.22 carefully and you'll pick up some good tips to make your drawings look really professional.



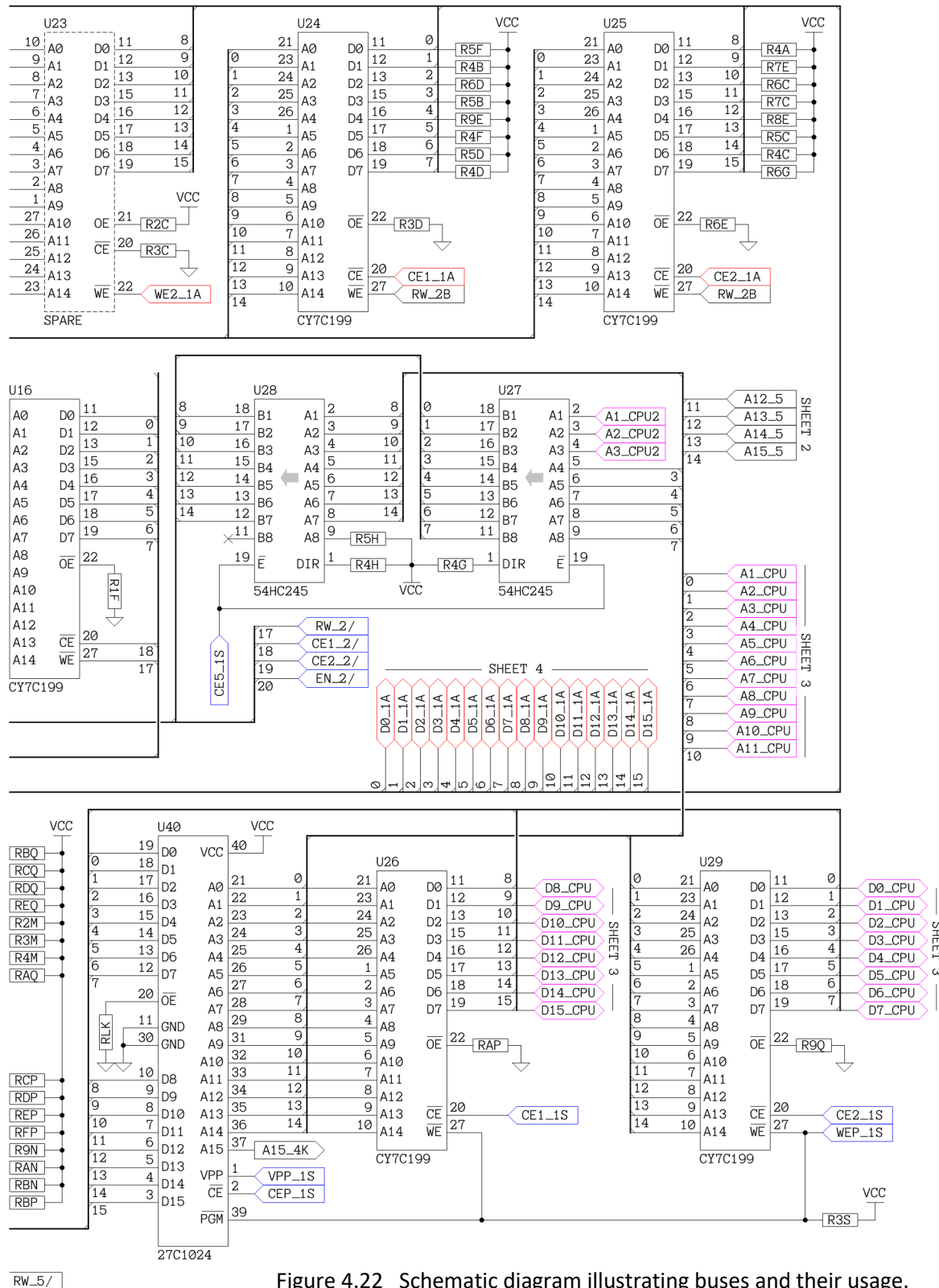


Figure 4.22 Schematic diagram illustrating buses and their usage.

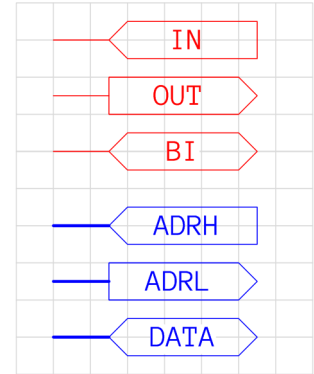
Signals and Labels

Signals are an integral part of a schematic diagram and labeling or naming them appropriately is key to properly documenting and understanding the circuit (e.g. Figure 4.6). We have briefly talked about labeling the signal pins of IC symbols earlier; now let's discuss a little about another type of labels known as signal connectors.

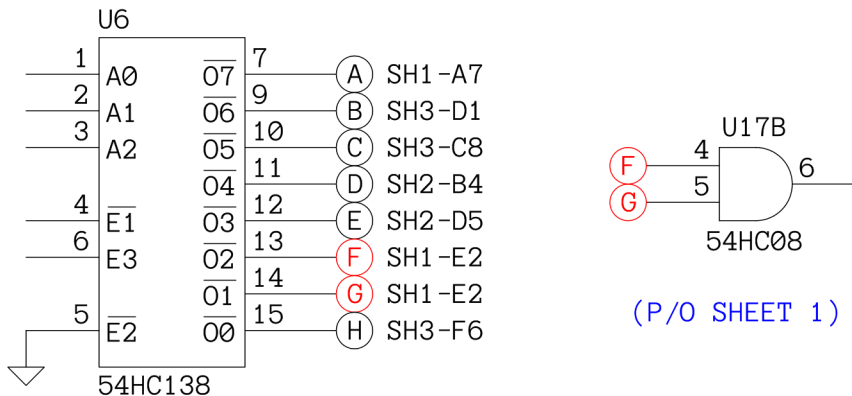
Basically, there are three types: Input, Output and Bi-directional. And these can be individual signal connector (red) or a group of signal connectors such as a signal bus (blue).

The main uses of these signal connectors are:

- To connect components that share the same network across multiple sheets, and
- To identify the directional flow of signals.



Sometimes a circuit's connectivity can be quite complex that it becomes difficult or cumbersome to assign signal names. This is especially true of PCBs with many PLD chips that function as decoding logic. Naming every output pin of these 'black boxes' becomes impractical. This is where another form of connector comes in quick and handy—the bubble connector:



The figure above shows a 3-to-8 decoder logic with eight outputs linked to bubble connectors A to H, and each bubble is associated with a sheet number and a reference grid coordinate of that sheet to enable quick location of the connecting point, for example F and G (red) are found in sheet 1 coordinate E2 which are the inputs of an AND gate (54HC08).

Now that we've covered the basics of schematic symbols and how they're wired up in a circuit, let's move on to the approach and strategy of reverse engineering PCBs.

APPROACH AND STRATEGY

One of the biggest mistakes that a person can make when doing PCB reverse engineering is to jump straight in and start tracing the connections and transferring them onto the electronic CAD worksheet. Without careful planning or a proper strategy, the initial enthusiasm will soon lead to frustration and loss of focus, and result in wasted time and energy. What then is the approach? The answer lies in the well known phrase: **divide and conquer**. Simply put: know what you want to do, why you are doing it, and where to look for the likely answers.

Having said that, the types of PCBs are so vast and the kind of functions they performed so varied, that to give a complete treatment of each type would be impossible, if not impractical here. A sample listing of these PCBs by types and functions are given in the table below:

Table 4.2 PCBs Types and Functions

Types	Functions	
Analog	Power Supplies	Linear regulator
		Switch-mode SMPS
		AC-DC, DC-DC, UPS
	Amplifiers	Audio – AM, FM, PCM
		Video – PAL, NTSC
		RF – HF, VHF, UHF
	Filters	Active, Passive (LP/HP/BP)
		Linear, Non-linear
	Servos/Motors	Control, PWM
		Encoder/Decoder
		Drive module
Digital	Processors	Microprocessor (CPU)
		Microcontroller (MCU)
		Signal processor (DSP)
	Memories	Volatile (SRAM, DRAM)
		Non-volatile (ROM, Flash)
	Logical	Combinatorial (Boolean)
		Sequential (Sync, Async)
Mixed	Signal Converters	Analog-to-digital (A/D)
		Digital-to-analog (D/A)
	Digital filters ¹⁴⁸	Recursive (IIR)
		Non-recursive (FIR)
	Instrumentation	Sensors, Transducers

¹⁴⁸ Digital filters operate on sampled, discrete-time digitized signals whereas analog filters operate on continuous-time analog signals. It's categorized under mixed type PCBs because implementing digital filters require the use of ADCs and DACs. Digital filters are categorized under recursive (infinite impulse response, IIR) and non-recursive (finite impulse response, FIR) types.

Chapter 4

We will look at a purely digital PCB as an example—the SCSI PC host adapter. The approach and strategy will vary for different types of PCB but the idea here is to get you started and thinking so you will be able to improvise and formulate your own strategies for the kind of PCBs you will work on in the future.

Here is the complete parts list again for your reference:

U1	SN74LS688N	U9	SN74LS688N	C1	100nF	C9	100nF
U2	DIP Switch	U10	74LS244N	C2	100nF	C10	100nF
U3	SN74LS04N	R1	1000	C3	12.6nF, 6V	C11	100nF
U4	DM74LS138N	RP1	007-6718203	C4	100nF	C12	47uF, 6V
U5	SP8924	RP2	007-6718203	C5	100nF	C13	47uF, 6V
U6	SP8918	RP3	007-6718203	C6	100nF		
U7	NCR-5380	F1	1.5A, 125V	C7	100nF		
U8	D2764A	CR1	1N5817	C8	100nF		

The pin-outs of common TTL ICs (marked in red) can be found in [Appendix D-2](#). U5 and U6 are obsolete parts and no information or datasheet can be found online about them. However, all is not lost.¹⁴⁹ I have place their partial pin-outs together with that of U7 and U8, along with the pin-outs for the ISA bus and the Centronics SCSI connectors (see overleaf).

Fortunately, the PCB we're going to work on does not have conformal coating, so we have one less task to do. You will need a reliable multimeter that has the diode test function which emits a beep sound when the probe leads are shorted together. It is also helpful to have a PCB layout diagram (draw one if it is not available) for you to highlight the pins that are validated so you will not waste time going through them again. This will minimize the amount of probing and prevent possible damage to the PCB as a result of repetitive probing or scratching activities.

You can use the PCB layout diagram on [page 91](#) if you have followed the exercises in [Chapter 3](#). I have the habit of drawing both the component and solder sides of the PCB layout diagram, sometimes out of necessity, especially for surface-mount PCBs where SMD components are found on both sides anyway, and sometimes because certain components obscure their footprints and makes it difficult for me to highlight or color the pins that have been validated. It's a matter of personal choice as to whether you think it's worth expending that additional effort.¹⁵⁰

¹⁴⁹ While you may think it's pointless to reverse engineer a PCB containing obsolete parts, let me remind you that PCBs with obsolete parts as well as proprietary or customized parts are quite common, especially in military PCBs. I have personally worked on many such PCBs and can confidently say that less than 10% of the failures associated with these PCBs involved those parts.

¹⁵⁰ It's really not difficult to draw the solder side of a through-hole PCB since they consists of mainly solder pads of the component footprints (ICs, transistors, etc.), which can be quickly drawn using the [Array Shapes...](#) tool and then align them over their symbols on the component side. Once done, simply group the whole PCB and do a [Flip Horizontal](#), then delete the symbols but keep the solder pad footprints. For axial leaded parts, first ungroup them, then delete the component bodies, and leave the solder pads as they are. One last thing is to delete the SCSI bulk connector mounting body and put in place two screws—and you are done!

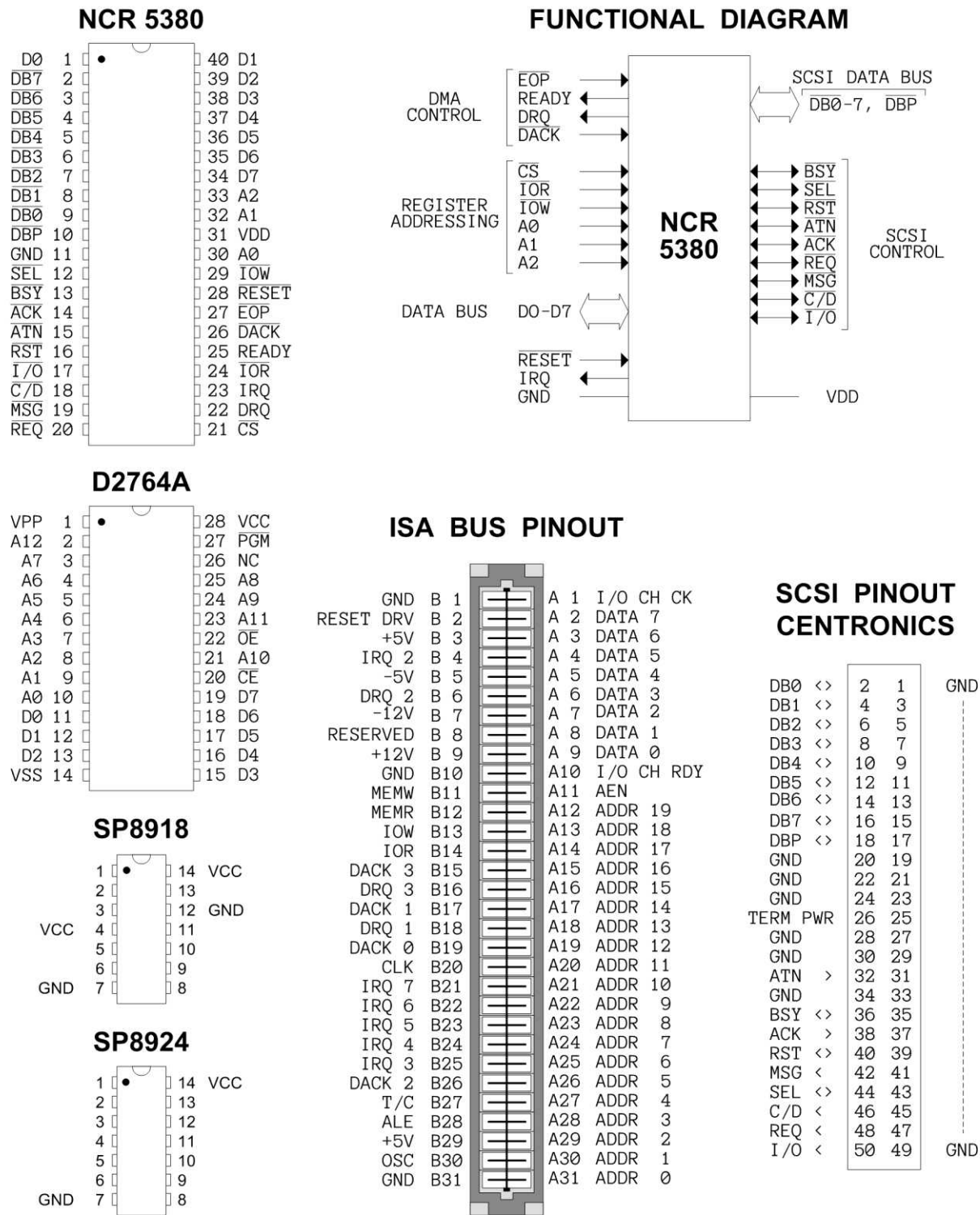


Figure 4.23 Pin-out information of the various components.

Here's the solder side layout of the SCSI host adapter:

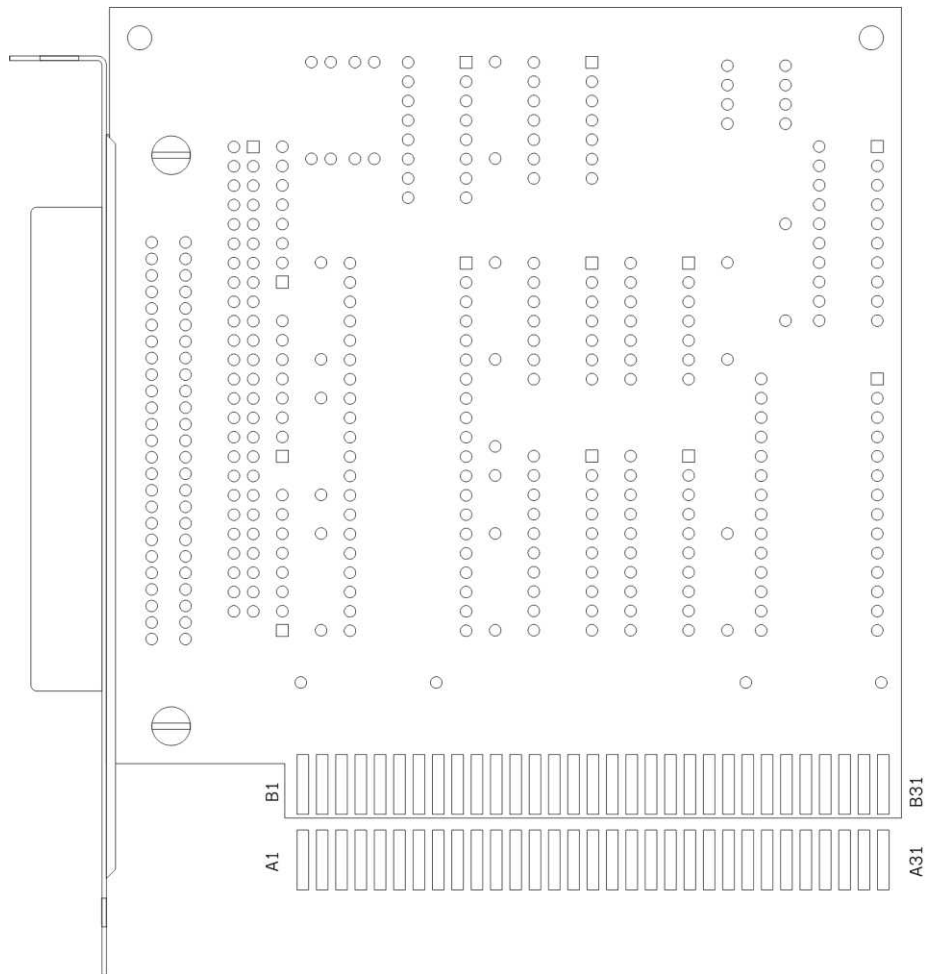


Figure 4.24 The solder side layout of the SCSI host adapter card.

Notice that I've substituted pin 1 of all the ICs, resistor networks and the 50-pin header (J1) with a square to facilitate ease of referencing while doing validation and highlighting concurrently. Also the ISA bus gold finger edge connector has been un-filled for coloring purpose, and the component side (A) is extruded next to the solder side (B) as well.¹⁵¹

[Statistics](#)

If you do a pin count, there are 350 solder pads (including the header and SCSI connector) and 62 edge fingers, giving a total of [412](#) probing points, which is not a lot (thankfully) but should be sufficient to serve our purpose in demonstrating the strategy which I will be covering next.

¹⁵¹ It took me less than half an hour to produce the solder side layout of the PCB, including a few simple touch-ups, so I'm sure you should not have any problem duplicating my effort either!

1. Validate all power and ground connections

A digital PCB will at least have for its power source a +5Vdc known as VCC and a reference called GND. This power source will connect to all the logic components on the PCB, usually the VCC to the highest numbered pin and the GND to the opposite mid-numbered pin of the component.¹⁵² For example, a 14-pin 7400 quad two-input NAND gates IC will have pin 14 for VCC and pin 7 for GND.

So the first step is to validate that all the components have their power pins connected to these two references, including decoupling capacitors. If pull-up resistors or resistor networks are present, check if the common end is connected to VCC or GND too. At the same time, it would be useful to look out for signal pins that are directly tied to either of them, since it is common practice for logic components to have certain of their inputs permanently set high or low.

With this in mind, let's start validating the power and ground pins based on the pinouts. Here is the net listing:

```
VCC  U1.20 U3.14 U4.16 U5.14 U6.14 U7.31 U8.28 U9.20 U10.20

VCC  C1.1 C4.1 C5.1 C6.1 C7.1 C8.1 C9.1 C10.1 C11.1
      C12.1 C13.1

VCC  P1-B3 P1-B29

GND  U1.10 U3.7  U4.8  U5.7  U6.7  U7.11 U8.14 U9.10 U10.10

GND  C1.2 C2.2 C3.2 C4.2 C5.2 C6.2 C7.2 C8.2 C9.2
      C10.2 C11.2 C12.2 C13.2

GND  J1.1 J1.3 J1.5 J1.7 J1.9 J1.11 J1.13 J1.15 J1.17
      J1.19 J1.20 J1.21 J1.22 J1.23 J1.24 J1.25 J1.27 J1.28
      J1.29 J1.30 J1.31 J1.33 J1.34 J1.35 J1.37 J1.39 J1.41
      J1.43 J1.45 J1.47 J1.49

GND  J2.1 J2.3 J2.5 J2.7 J2.9 J2.11 J2.13 J2.15 J2.17
      J2.19 J2.20 J2.21 J2.22 J2.23 J2.24 J2.25 J2.27 J2.28
      J2.29 J2.30 J2.31 J2.33 J2.34 J2.35 J2.37 J2.39 J2.41
      J2.43 J2.45 J2.47 J2.49

GND  P1-B1 P1-B10 P1-B31

VCC  U1.4 U1.6 U1.13 U1.15 U4.5 U6.4 U8.1 U8.27 U9.2
      U9.6 U9.13 U9.15

VCC  R1.1

VCC  CR1.AN

GND  U1.2 U1.8 U1.11 U1.17 U1.18 U6.12 U8.20 U9.1 U9.4
      U9.8 U9.11 U9.17

GND  RP1.1 RP2.1 RP3.1
```

¹⁵² There are exceptions such as the NCR-5380 with its power at pin 31 and ground at pin 11. Some ICs may even have their power and ground reversed from the TTL standard orientation, such as the 1,048,576 x 4 dynamic RAM TC514400, for example.

Statistics

From the net listing, we see the power and ground pins made up 148 out of 412 or 35.9% of the total solder pads or pin counts on this PCB. This is, of course, attributed to the fact that the SCSI pinout alone contributed a substantial amount to the total. If we were to remove this factor or 50 pins off the record, which is 98 out of 362 or 27.1% of the PCB pin count, it would be an impressive figure still.

On a more conservative estimate, we can estimate that as much as 20% of a digital PCB may be covered by just validating these power and ground pins alone—not bad for a start to get your morale on high gears!

Note that the first group of three VCC and five GND nets are the standard power and ground pins for the components and connectors. The second group of three VCC and two GND nets are the signal pins, pull-ups and the diode CR1 that supplies the termination power to the SCSI connector.

To complete the rest of the power-related nets:

```
VTP CR1.CA C2.1 C3.1 F1.1 RP1.8 RP2.8 RP3.8  
F2.2 J1.26 J2.26 (TERM PWR)
```

We can see from both net listings that RP1 thru RP3 are dual termination resistor network that are used in SCSI termination network, but we'll leave that to the next section. One thing to bear in mind is that with the proliferation of low-power digital ICs to achieve greater clock speeds, digital PCBs no longer operate just on the standard +5Vdc now. PCBs with new designs will likely have circuits that operate on 3.3V, 2.5V, 1.8V, and even 1.5V supply voltages.¹⁵³ It helps to tabulate the ICs according to their power type as you gather datasheets and information as detailed in [Chapter 2](#).

2. Validate address and data buses

If the PCB you are working on contains microprocessor, memory and peripheral components, check for connectivity on the address and data buses based on the datasheets of these components. It is good to start from the data bus since most bused components share the same data lines (i.e. D0–D7). Next, check the address bus, starting from the lower address lines (i.e. A0, A1, A2 ...) because the higher address lines are usually used for decoding purposes and is less likely to be shared.

From the component pinouts, we can gather that both U7 (NCR-5380) and U8 (2764A) contain data and address elements, as well as the J1 and J2 SCSI connectors, and the ISA bus edge connector. But we are not discounting the other ICs, since there is a 74LS138 (U4) which is a 3-to-8 decoder chip that is commonly known to connect to address lines also.

¹⁵³ Look out for the presence of low dropout regulators such as the LM3940 or LM1117 which convert the +5Vdc supply to the +3.3V, +2.5V, +1.8V or +1.5V outputs. This is a good indication that the PCB contains low-power ICs.

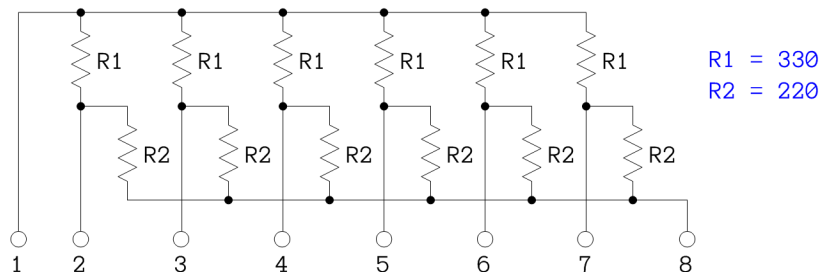
Here's the ISA bus net listing:

```
D0  P1-A9 U7.1  U10.3
D1  P1-A8 U7.40 U10.5  U6.2
D2  P1-A7 U7.39 U10.7
D3  P1-A6 U7.38 U10.9
D4  P1-A5 U7.37 U10.12
D5  P1-A4 U7.36 U10.14
D6  P1-A3 U7.35 U10.16
D7  P1-A2 U7.34 U10.18
```

```
U8.11 U10.17
U8.12 U10.15
U8.13 U10.13
U8.15 U10.11
U8.16 U10.8
U8.17 U10.6
U8.18 U10.4
U8.19 U10.2
```

```
A0  P1-A31 U4.1  U7.30 U8.10
A1  P1-A30 U4.2  U7.32 U8.9
A2  P1-A29 U4.3  U7.33 U8.8
A3  P1-A28 U1.3  U8.7
A4  P1-A27 U1.5  U8.6
A5  P1-A26 U1.7  U8.5
A6  P1-A25 U1.9  U8.4
A7  P1-A24 U1.12 U8.3
A8  P1-A23 U1.14 U8.25
A9  P1-A22 U1.16 U8.24
A10 P1-A21 U8.21
A11 P1-A20 U8.23
A12 P1-A19 U8.2
A13 P1-A18 U9.3
A14 P1-A17 U9.5
A15 P1-A16 U9.7
A16 P1-A15 U9.9
A17 P1-A14 U9.12
A18 P1-A13 U9.14
A19 P1-A12 U9.16
```

From the net listing, we see that the ISA address and data lines go to U7-U10, with the data bus going to the EPROM (U8) through the octal buffer (U10). Notice that A0-A2 goes to the decoder U4. The other set of data lines from the SCSI controller chip (U7) will go to the SCSI connectors J1 and J2, but each line must be properly terminated by a 220/330 ohms resistor pair, which is provided by the resistor packs, RP1 and part of RP2. As with most communication protocols, the parallel SCSI data bus also has a parity bit line known as the Data Bit Parity (DBP).



Chapter 4

And here's the SCSI controller's data bus net listing:

```
DB0  J1.2 J2.2 U7.9 RP1.7
DB1  J1.4 J2.4 U7.8 RP1.6
DB2  J1.6 J2.6 U7.7 RP1.5
DB3  J1.8 J2.8 U7.6 RP1.4
DB4  J1.10 J2.10 U7.5 RP1.3
DB5  J1.12 J2.12 U7.4 RP1.2
DB6  J1.14 J2.14 U7.3 RP2.7
DB7  J1.16 J2.16 U7.2 RP2.6
DBP  J1.18 J2.18 U7.10 RP2.5
```

Statistics

The address and data buses made up 126 out of 412 or 30.6% of the total pin counts on this PCB. Discounting the SCSI ground pins, we have 126 out of 362 or 34.8%.

In general, address and data lines make up approximately 20-40% of a digital PCB, depending on the design and components used. This means that by just working on these two validation points, you can easily cover between 40-60% of a digital PCB! Adopting the divide and conquer strategy does cut down effort and save time.

3. Validate known signals

Next, you should start working on known signal names i.e. connector or IC with signal names, such as the reset (RST), chip select (CS), read (RD), write (WR), output enable (OE) signals, etc. Most of these are control signals and should not be too difficult to locate them since the pinouts will indicate the pin numbers and if functional diagrams are available, the direction flow of these signals.

Here are the remaining ISA bus connector pins:

```
AEN  P1-A11 U1.1
RSTDRV P1-B2 U3.3
      U3.4 U6.1 U6.13 U7.28 (/RESET)
IRQ2  P1-B4 U7.23 (IRQ)
MEMR  P1-B12 U9.18
IOW   P1-B13 U4.4 U7.29 (/IOW)
IOR   P1-B14 U7.24 (/IOR)
DACK3 P1-B15 U2.5
DRQ3  P1-B16 U2.6
DACK1 P1-B17 U2.7
DRQ1  P1-B18 U2.8
T/C   P1-B27 U3.5
      U3.6 U7.27 (/EOP)
```

Two of the signals, RSTDRV and T/C, go through the inverter (U3) before driving the inputs of U7. You should try to group the inverter outputs with its inputs to complete the flow of the signal, if possible. Note the signal names of U7 pins (green) are put in braces for references, and the '/' before a signal name denotes an active low signal.

Here are the remaining SCSI connector pins:

```

ATN  J1.32 J2.32 RP2.4 U7.15 (/ATN)
BSY  J1.36 J2.36 RP2.3 U7.13 (/BSY)
ACK  J1.38 J2.38 RP2.2 U7.14 (/ACK)
RST  J1.40 J2.40 RP1.7 U7.16 (/RST)
MSG  J1.42 J2.42 RP1.6 U7.19 (/MSG)
SEL  J1.44 J2.44 RP1.5 U7.12 (/SEL)
C/D  J1.46 J2.46 RP1.4 U7.18 (/C/D)
REQ  J1.48 J2.48 RP1.3 U7.20 (/REQ)
I/O  J1.50 J2.50 RP1.2 U7.17 (/I/O)

```

You can see that it's quite straightforward once you have the pinouts of the connectors and components ready on hand. The rest of the known signals are:

```

U3.1 U5.2 U7.22 (DRQ)
U5.6 R1.2 U7.26 (/DACK)
U1.19 U4.5 U7.21 (/CS)
U9.19 U10.1 U10.19 U8.22 (/CE)

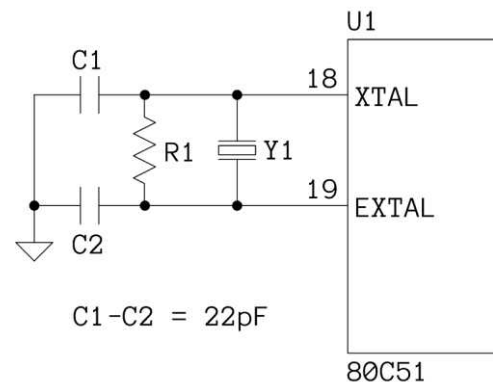
```

I did not include inverter output U3.2 with its input U3.1, but intentionally put it under the unknown and hidden traces section later as an example. This is after all a small PCB and I have to make full use of whatever limited resources available to make the most of the lessons I want to impart to my readers.

Some known signal names are rather easy to link together, especially in a CPU-based PCB where many ICs share the same signal notations (e.g. RD, WR, CS, RST, etc.), but some will require more effort to decipher and pair up with related discrete components. One example is the crystal oscillator circuit that is used to clock a processor chip.

Consider the Intel 80C51 microcontroller. If you refer to its datasheet, there is a section outlining the recommended clock design specifications, and most PCBs that employ these chips will most certainly follow what the datasheet says, so look out for components shown on the right (Y1, R1, C1 and C2) near the IC's vicinity.

More complicated CPU boards may contain voltage powered oscillator ICs with TTL outputs driving counter circuits to sub-divide its frequency into lower values before driving the CPU and its peripheral ICs.



4. Validate jumper connections

These are the next in line candidates to consider since most of the time they are located close to the components which they are connected to. However, care must be exercised not to remove the jumpers without first taking note of their positions. That's why a PCB layout diagram comes in handy since any jumper will have its position indicated (refer to Figure 3.1).

In the case of the ISA bus SCSI host adapter, there is no jumper but a DIP switch and the positions of its four rocker switch levers are also indicated in Figure 3.13.

U2.1 U2.3 U5.3 (DRQ1 | DRQ3)
U2.2 U2.4 U5.5 U6.11 (DACK1 | DACK3)

Switch U2 provides selection of either DRQ1/DACK1 or DRQ3/DACK3 pair to the unknown U5, so we now know two of its signal pins are DRQ and DACK related.¹⁵⁴ DACK1 or DACK3 also goes to unknown U6 pin 11.

5. Validate visible traces

Most PCBs in existence are likely to be multilayer type, i.e. they have many stacked layers in which the electrical traces are routed and interconnected by means of [through](#) and [hidden](#) vias. Traces found on the component side and solder side of the PCB are visible to the eyes whereas those found in the inner stacked layers are hidden—but they are nevertheless there.

After covering the five areas as detailed above, the PCB Layout diagram should have quite a substantial portion marked as validated. The remaining unmarked pins of the components are the ones left to be validated. It is, therefore, logical to start with those traces you can see and leave the hidden ones to the last. But in the process of doing so, always check against the component datasheets to ensure that any man-made mistakes are detected early.¹⁵⁵

The following nets are mainly from the two unknown ICs and therefore do not have known signal names. Fortunately, the PCB traces from their pins are visible on both sides and therefore traceable.

U5.1 U6.6
U5.4 U6.8
U4.13 U6.3

6. Unknown and hidden traces

The most difficult part is usually the remaining 3-5% of the PCB. This is often known as the point of diminishing returns, meaning that for the same amount of effort expended the results become lesser and lesser (very much like the charging capacitor analogy).¹⁵⁶ These are the most difficult and time-consuming to locate, especially if there are a lot of ICs (e.g. programmable logic devices, obsolete and customized ICs) with unknown signal names; but using a common sense approach, you may still be able to find half to nearly 90% of them!

The common sense approach is:

- Refer to datasheet for hint of usage

Some logic devices such as peripheral ICs have different functional modes and their signal pins must be connected in certain configurations. In some cases, certain pins must be left unused, unconnected, or pulled high or low.

¹⁵⁴ DRQ – Data ReQuest; DACK – Data ACKnowledge.

¹⁵⁵ Human fatigue is the primary cause for man-made mistakes, or carelessness such as wrongly counting the pin position or taking reference to wrong component orientations, especially from the solder side view.

¹⁵⁶ That is the reason why the three D's mentioned in [Chapter 1](#) are so important!

- Focus on un-validated pins

The PCB layout diagram identifies those pins that are still un-validated. This is where you should focus your effort on to reduce the pin count.

- Obey electrical rules

Output pins should not be connected together unless they are open-collector types. Multiple input pin drives may indicate a need for buffering.

- Refer to the PCB's functional block diagram

Sometimes functional block diagram of the PCB may be available and this is a very great source of information in deciphering the topology of the PCB itself.

- Refer to system interconnection diagram

Sometimes the system interconnection diagram which the PCB is a part of may be available. This can be helpful in determining which edge connector pins are used and which are not used.

(Note—If all else fail, refer to the last section of this chapter!)

There are a total of 24 solder pads and 18 edge connector pins left un-validated. Discounting non-connect (NC), reserved and unused power pins, we have a total of 37 out of 412 or 9%. And after going through three rounds of validation, only one net is found:

U3.2 U6.10

7. Unused pins

For a small PCB like the ISA bus SCSI host adapter, having 35 unused pins is a high percentage (8.5%) of the total pin count, but bearing in mind that it only uses two-thirds of the ISA bus signals for its limited functionality, that is still an acceptable figure.

So how does the schematic diagram look like? Turn overleaf for the answer.

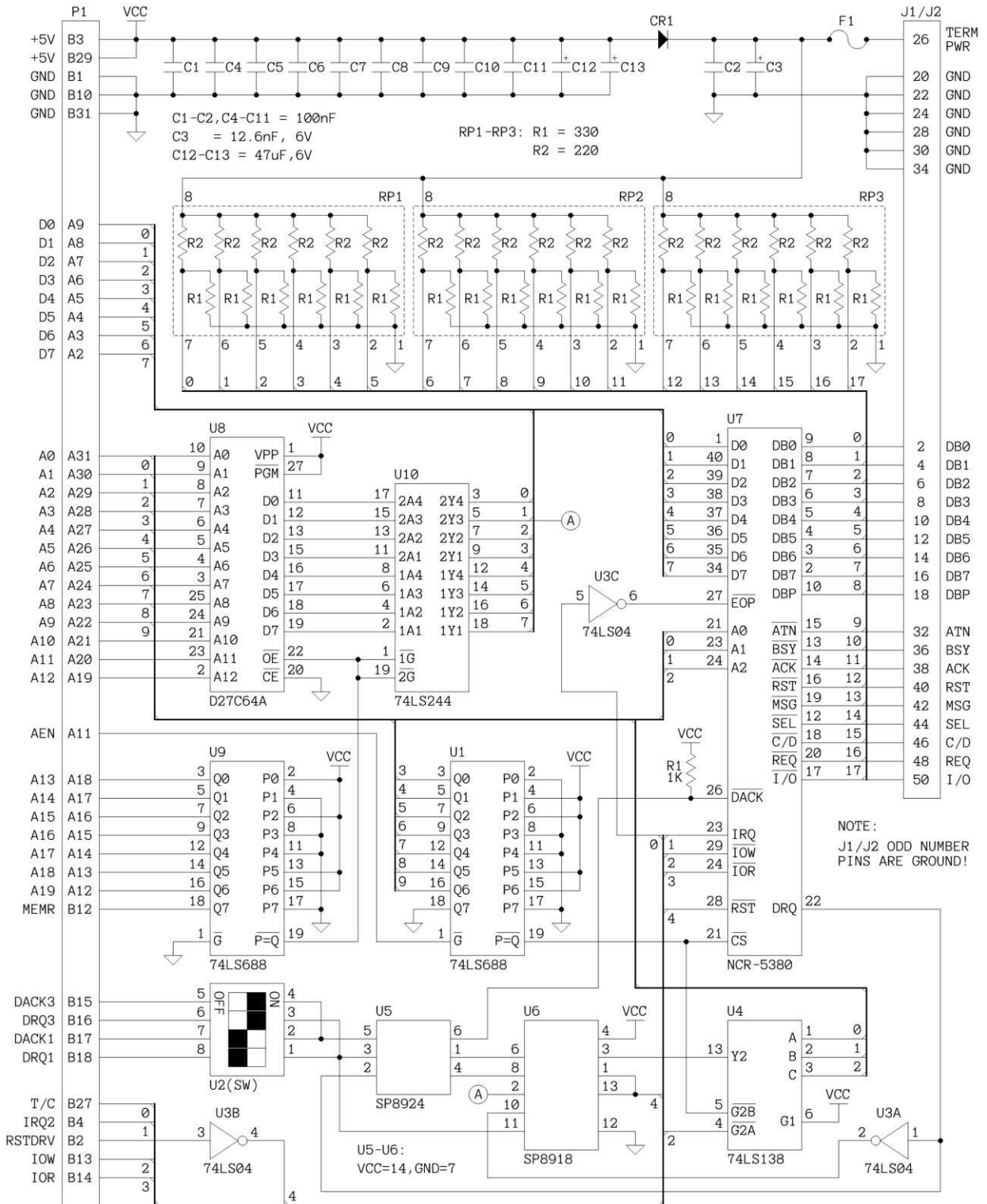


Figure 4.25 Schematic diagram of the SCSI host adapter card.

Believe it or not, reverse-engineering work for this PCB only commenced as I started writing this portion of the book. Incredibly, I managed to squeeze the whole schematic diagram into a single page of this book,¹⁵⁷ and am quite pleased also to be able to implement a number of elements which I've mentioned in the earlier sections of this chapter into the works. Do take the time to study the schematic diagram and re-read [Elements of a Good Schematic Diagram](#) on [page 131](#) again.

It's a pity I could not find the datasheets for U5 (SP8924) and U6 (SP8918) online or on any of the archive websites containing scanned collections of old National Semiconductor data books, which means the ICs will remain as black boxes in the schematic diagram for now.

Here's the PCB layout diagram with the validated pins highlighted accordingly, just to add icing to the cake:

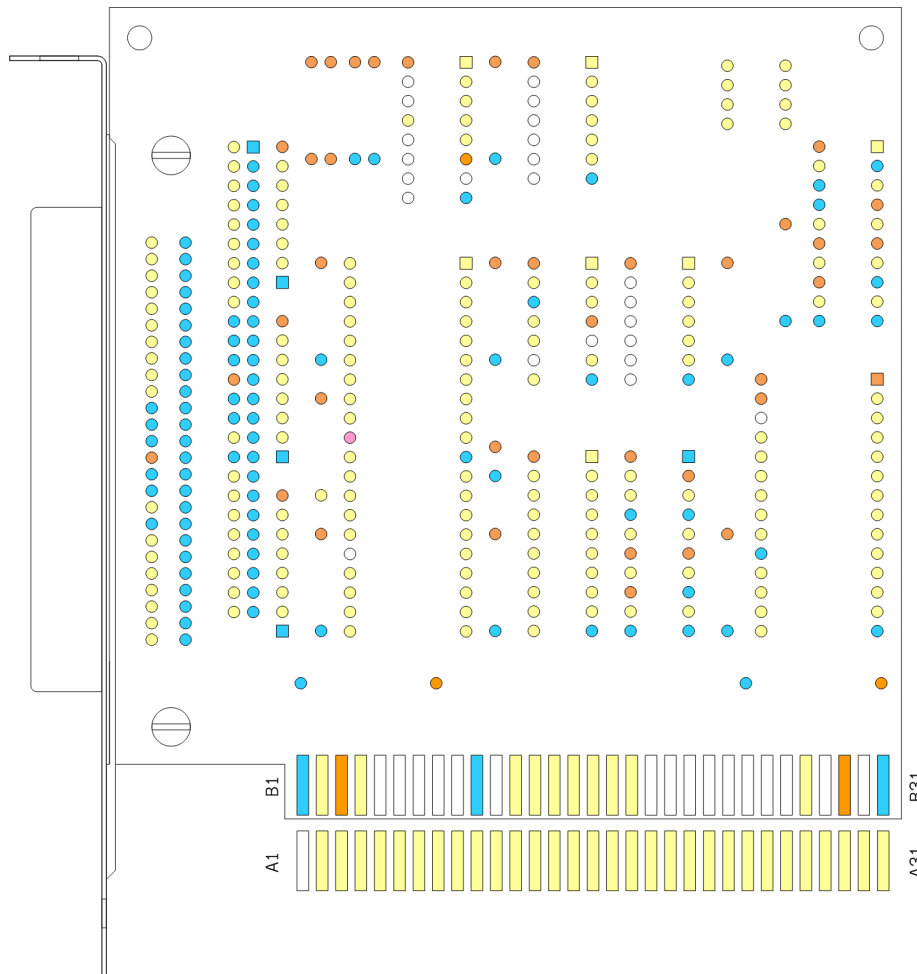


Figure 4.26 PCB layout diagram with validated pins highlighted.

¹⁵⁷ It took me about two-third of a day to complete the validation and create the net listing, and another one and a half day to draw and edit the schematic diagram to give it the optimum placement and signal flow.

Finishing a reverse-engineering job always brings a sense of satisfaction and achievement, even if the candidate happened to be a simple board. Now that we've completed a digital PCB, let's take a look at an analog PCB for a change.

IT'S AN ANALOG WORLD!

We live in an analog world that is fast becoming digitized as digital technology continues to invade our homes and offices. However intrusive this digital revolution may be, the realm of the discrete or finite cannot effectively interact in a meaningful way with the real analog world without the help of sensors or transducers, which are themselves, analog.

Analog PCBs are a different breed compared to their digital counterparts in terms of their design process and board layout requirements. This means that the aforementioned approach and strategy for digital PCBs cannot be applied wholesale on analog PCBs per se. To complicate things further, different types of analog PCBs will require slightly different approaches to handle their validation processes.

The good news is analog PCBs are not as complex and dense as digital PCBs, by reason of the need to adhere to strict design rules to ensure signal integrity, avert distortion and reduce inherent parasitic noise to a minimum.¹⁵⁸ Generally, analog PCB designs have not made great strides like their digital cousins in terms of complexity and scale, but they are much harder to design, because of the subtle analogue nature of the components and circuits alike, and if not given careful design considerations, could vastly affect the performance and outcome of the intended design functionalities.¹⁵⁹

One of the key points to remember when doing reverse-engineering on an analog PCB is to [take note of the PCB's circuit topologies](#). What this implies is that within an analog board you will find a broad range of circuits such as power supplies, filters, operational amplifiers,¹⁶⁰ etc. each with their own topological make-up that defines the purpose or function for that part of the PCB.

Knowing what to look for will greatly reduce the effort to link up the related components for that part of the circuit, and will further validate that you are doing it right. Therefore identifying an operational amplifier IC and its surrounding discrete components, such as the presence of capacitors and resistors may point to an integrator or differentiator, or a clipper or limiter in the case of diodes. That's also the reason why creating a BOM list and obtaining component datasheets during the preparation phase are important and helpful.

¹⁵⁸ An analogy can be made about the difference between analog and digital by inferring to a man and a woman. Man is digital because he has a logical composition, while a woman is analog because she is more sensitive and has a wider range of emotions.

¹⁵⁹ No two analog PCBs are the same—even if they are similar in design and use the same batch of components or assembly process. That's why you are more likely to encounter components marked as FSV on an analog PCB than a digital one. FSV stands for factory selected value.

¹⁶⁰ I've made a brief reference to op amp circuit configurations on [page 149](#) while discussing on how to draw their schematic symbols.

AN ATX POWER SUPPLY EXAMPLE

After giving some considerable thoughts, I've decided on an ATX power supply for the analog example for the following reasons:

- Power supplies are very common, whether as standalone modules or as part of a PCB, and the probability of you coming across one is definitely much higher,
- Switch-mode power supplies (SMPS) are gaining popularity and preference in use for their good power efficiency and smaller sizes compared to a linear one, and
- ATX power supplies are readily available.

In the course of my work, I've repaired quite a number of industrial PCs used in military systems. Most of the time, the failures are attributed to two components: the power supply and the motherboard. Commercial PCs usually last no more than seven years before they start to breakdown, and again the power supplies are usually the first to go.¹⁶¹

Rummaging through the pile of defective power supplies, I found a suitably simple model that uses the SG6105 power controller chip made by System General Corp. I searched and downloaded the datasheet, and came across a suggested detailed example circuit in the application notes section (see overleaf Figure 4.26).



Instead of going head-on to reverse-engineer the power supply unit, I went online again to see if there's any example of a similar wattage model with ready schematic diagram. Indeed there are plenty of them out there, some of which are the works of individuals like Vladimir Davydov and Pavel Ruzicka. This goes to show that there are engineers in different parts of the world who, like me, have the same pursuit and passion in reverse-engineering work.

After going through quite a number of the schematic diagrams, I opted for the model built by M-Tech. I've intentionally placed the datasheet's suggested example and the real-world implementation side-by-side for direct comparisons. You'd notice that they're both quite similar, except that the real-world example has enhancements and other extras than the datasheet's. It also proved a point I mentioned in the last part of [Chapter 2](#) under [Component Datasheets](#) on [page 48](#), that datasheets may provide useful design application notes that give good hints on how a component can be used in relation with other components, which will certainly cut down time and effort in your work.

¹⁶¹ My six-year old home PC developed a strange behavior sometime early this year. It would intermittently freeze upon power up as the Gigabyte motherboard runs through the POST and could not complete. There was not the usual beep sounds to indicate the source of the problem, and after some fiddling and swapping of RAM sticks and video cards, I finally suspected the power supply to be the culprit. After replacing the old 450W SMPS with a brand new 500W one, the problem disappeared.

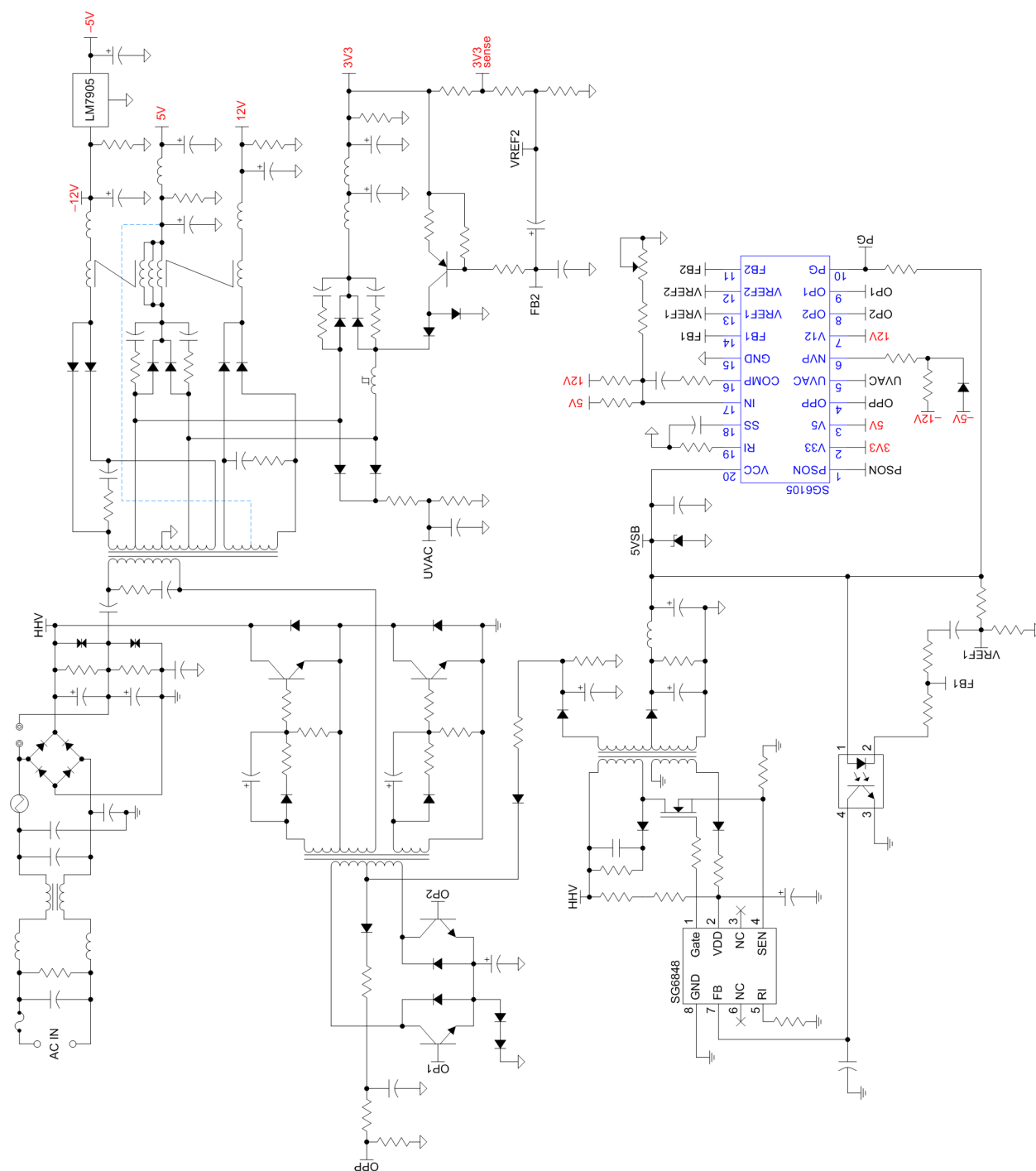


Figure 4.27 Suggested application example in the SG6105 datasheet.

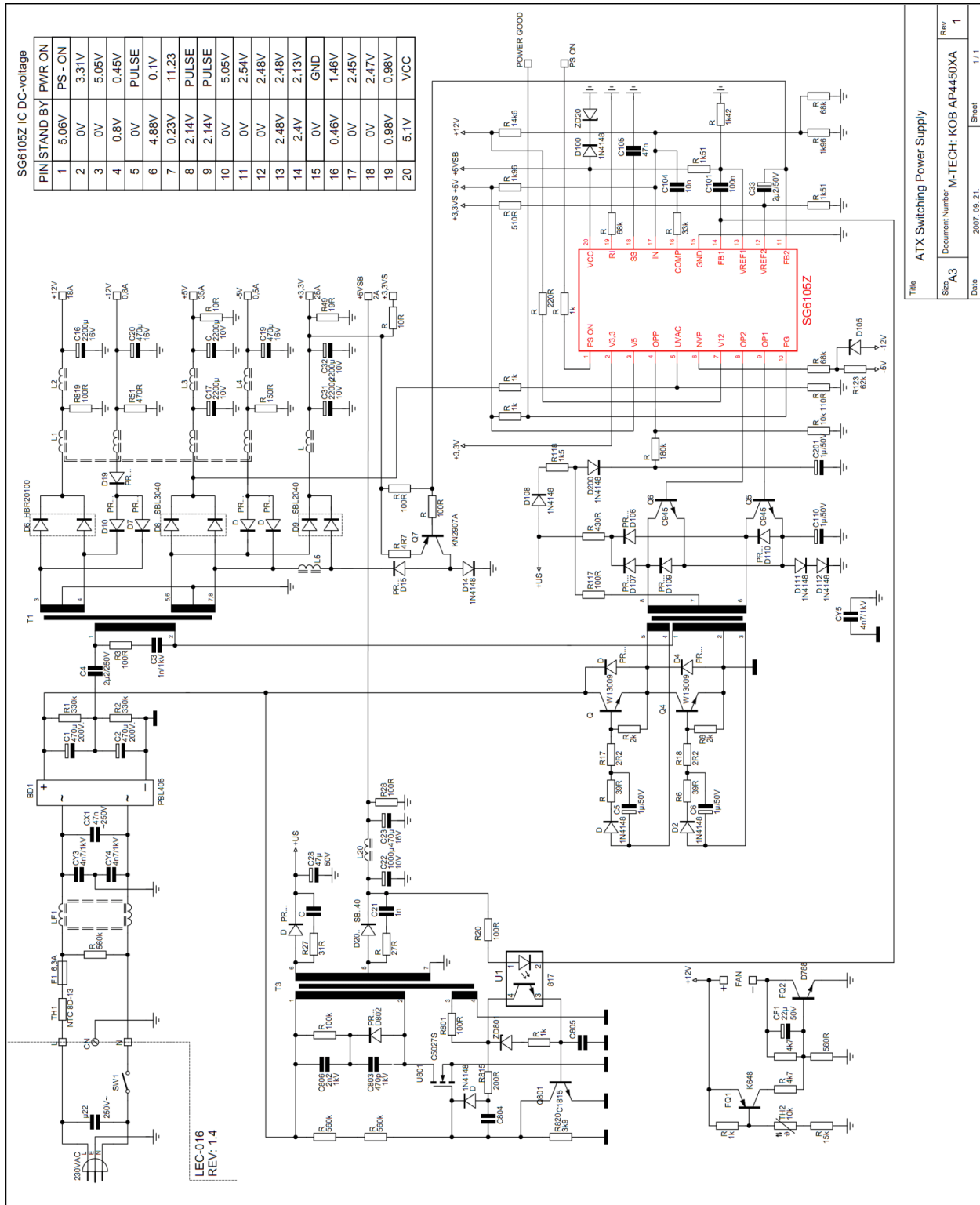


Figure 4.28 A 450W ATX power supply using the SG6105 chip.

STRATEGY FOR ANALOG PCBS

Though I mentioned that the approach and strategy for a digital PCB cannot be applied in totality on an analog PCB, the **divide and conquer** principle is still fundamental to the success of reverse-engineering any kind of PCB. The steps for validating an analog PCB are outlined as follows:¹⁶²

1. Validate all power and ground connections

Analog PCBs usually operate on more than one power source. Common supply voltages include +28V, $\pm 12V$, $\pm 15V$, +9V, and +24V for DC, and 240V, 115V and 26V in the case of AC. Generally, the number of power sources on an analog PCB can be inferred from its on-board components, which is why in the process of obtaining datasheets and information on these parts, you would have easily identified them already.

There may also be more than one analog ground reference, for example, earth or chassis ground at the primary AC side and DC reference at the rectified secondary side of power supplies, as well as floating outputs. In a mixed signal PCB, the analog and digital grounds are usually separate to prevent noise from the high-speed switching of digital components from affecting the analog part of the circuit.

2. Validate low impedance paths and parts

One of the challenges of analog PCBs are the presence of low impedance paths and component parts that may be present in considerable quantities. Some of these elements exhibit near zero or short characteristics, such as fuses, current-sensing resistors, inductors or transformers—while others like low-value resistors and IC signal pins with low impedances—may mislead you into thinking there is valid connectivity when using the diode function of the multimeter to provide audio indications during the probing and validation process.

Creating a BOM as you analyze and layout the PCB helps to identify these potential false flags, especially the discrete components. However, sometimes it may become necessary to remove some or all of these components so you don't have to constantly glance at the multimeter display just to be sure it's a true short and not a low impedance reading.¹⁶³

The reason for validating these paths and parts upfront—in fact, preferably before validating the power and ground connections—is to root out unwanted distractions and reduce incidences of error due to lapses in concentration which may be the result of fatigue and frustration arising from putting up with the constant beeping all over the PCB while performing validation.

¹⁶² These steps are only general guidelines and by no means exhaustive, especially when applied to analog PCBs. Of particular note is point 3 which emphasizes the importance of knowing and recognizing the topologies associated with different types of analog circuit designs.

¹⁶³ Any impedance below 100 ohms will usually register a continuous beep similar to a short when using the DMM in the diode function mode.

3. Validate known configurations

Earlier on, I mentioned that one of the key points to remember when doing reverse-engineering on an analog PCB is to [take note of the PCB's circuit topologies](#). The topology of a circuit defines how its various components or parts are connected to each other to achieve a certain function, whether in series, parallel or a combination of both.

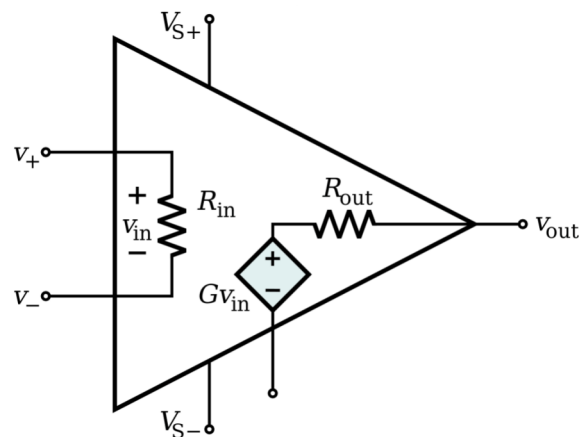
Digital circuits are broadly classified under sequential or combinatorial implementations; analog circuits, however, are more diversified. We will briefly take a look at three main types: operational amplifiers, power supplies, and filters.

Operational Amplifiers

Without a doubt, operational amplifiers are among the most widely used electronic devices today, and is considered to be one of the basic building blocks of analog circuits. Due to its almost ideal DC amplification properties, this linear device found extensive use in signal conditioning, filtering and mathematical function circuits.

[Appendix D-5](#) illustrates eight of the most common operational amplifier configurations, namely:

- Voltage comparator
- Voltage follower
- Inverting amplifier
- Non-inverting amplifier
- Integrator
- Differentiator
- Differential amplifier
- Inverting summing amplifier



These basic configurations are meant to serve as an introductory guide only. There are certainly other varieties such as the log amplifiers, oscillators, waveform generators, voltage references, active filters, audio and video amplifiers, etc. which are too numerous to list them all.

I would strongly recommend you get yourself a book on this subject, both as a quick reference and a good refresher. [Op Amp Applications Handbook](#) by Walt Jung of Analog Devices, Inc. is a hefty 900-page volume for the technically minded, while Texas Instruments' modest 94-page [Handbook of Operational Amplifier Applications](#) provides a quick fix and can be downloaded for free.¹⁶⁴ More importantly is to check out datasheets for application notes that provide practical hints on their usage, which in my experience, most manufacturers will usually include in their product specifications to ensure that designers use the recommended parts and properly layout the PCB footprints for optimum expected results.

¹⁶⁴ It might interest you that I have an international edition of the Prentice-Hall book [Op-Amps and Linear Integrated Circuits](#), Third Edition by Ramakant A. Gayakwad.

Power Supplies

In the early days, there is only one type of power supplies: the linear regulator. In its simplest form, the linear regulator comprises a bulk transformer core that steps down the input AC voltage to an acceptable level, which is then rectified by either a full-wave or bridge rectifier circuit and filtered by a large capacitor, before being voltage-regulated through a series power transistor via an output feedback control circuit.¹⁶⁵

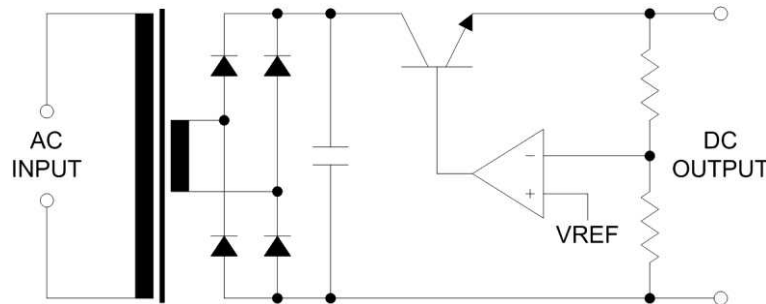


Figure 4.29 Functional diagram of a linear regulator.

Though the linear regulator is simpler in design, has better output regulation and is less noisy, it's many disadvantages such as being heavy and bulky, low efficiency and limited AC input range, etc. has led to the design of the switch-mode power supply (SMPS). The figure below shows the functional diagram of a simple SMPS:

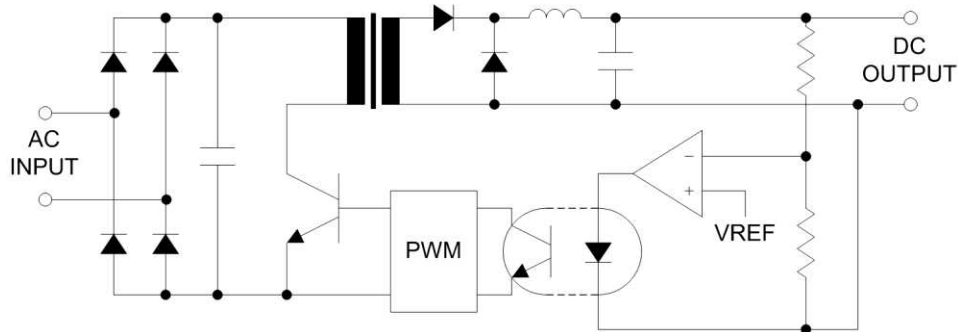


Figure 4.30 Functional diagram of a switch-mode power supply (SMPS).

Switch-mode power supply solves the size, efficiency and input AC range issues of the linear regulator, but is offset by its more complicated designs, requiring far more components to further address its inherent noise and regulation issues.

¹⁶⁵ This power transistor is also known as a pass transistor since the bulk of the load current passes through it. The output feedback control circuit is essentially made up of a sampling element such as a voltage divider that feeds the input of an error amplifier, which subsequently biases the pass transistor either directly or through a driver transistor. Incidentally, the linear power supply was also one of my second-year project in the polytechnic, the other being a manually operated digital IC-tester.

Different topologies of the SMPS can be found in [Appendix D-6](#), which are generally grouped into non-isolated and isolated types:

Non-Isolated SMPS	Isolated SMPS
Buck	Flyback
Boost	Forward
Buck Boost	Two-Switch Forward
Sepic	Active Clamp Forward
	Half Bridge
	Push Pull
	Full Bridge
	Phase Shift ZVT

There are many other derivatives of these topologies which you should be able to obtain information easily online. ON Semiconductor has a 73-page [Switch-Mode Power Supply Reference Manual](#) that deals in great details and comes with 16 pages of overview on the various topologies plus thirteen example circuits ranging from 10W to 500W designs. There's also a comprehensive list of design notes, tutorials and application notes at the end that you can reference, search and download from the manufacturer's website. These should provide sufficient pointers for your power supply reverse engineering work if you ever need it.

Filters

A filter is an analog circuit which processes a signal to remove undesirable frequency components while enhancing the desired ones. Filters can be classified as passive or active, analog or digital, time-discrete (sampled) or continuous, linear or non-linear. Types of filters include low-pass, high-pass, band-pass, band-stop (band-rejection; notch), or all-pass for analog filters, and infinite impulse response (IIR) or finite impulse response (FIR) for digital filters.

[Appendix D-7](#) illustrates the different types of passive filter circuits, and [Appendix D-8](#) to [D-9](#) contain the topologies of common active filters. In real world designs, filters are not limited to just the first- or second- orders. Critical circuits may employ cascading stages of these basic filter circuits to achieve up to the sixth or seventh order, so it is important to be able to recognize the fundamental designs to help you figure out the overall schemes, if you do come across one this complex.

Of course, each topology has its own range of characteristics, including actual (versus optimal) gain, the number of components used, gain bandwidth of operational amplifiers for active filters, relative noise, and the spread in RC component values. Though it is not necessary to know all these to do reverse engineering, having some background knowledge of the circuits you're trying to re-assemble does give you some advantage. This is the difference between a brute force reverse-engineering approach which totally disregard circuit topologies, and a strategized approach which is based on foreknowledge with an analytical mindset that constantly evaluates progress and correctness.

4. Validate visible traces

Analog PCBs are less complex than digital PCBs so naturally the number of layers employed in multi-layered designs will also be less in comparison. Some through-hole analog PCBs may simply be just double-sided which, if viewed through a bright light source, will even allow you to easily trace individual tracks from one point to another.

One useful piece of equipment I can think of is a light box that is commonly use for trace drawing. LED technology has matured to an extent it's now replacing conventional fluorescent lamp lighting in light boxes as the preferred light source, making it slim, light weight and energy efficient, as well as solving the problem of flicker that's associated with fluorescent tubes. In a sense, these new generation of light boxes can rightly be called light pads or light tablets, and you could easily get an A4 size model for less than \$50.



The advantage of using a light pad over lifting up the PCB overhead against a fluorescent light source is it gives a better contrast with an overall consistent brightness, besides sparing you the agony of a neck ache from repeatedly lifting up the PCB to trace the track while at the same time stabilizing your gaze. I've actually tried that several times with a couple of analog PCBs and I can tell you it's not only tiring for the eyes and neck, often errors are made as a result of fatigue and strain. Besides, the light pad is useful for any PCB that does not contain power or ground planes that obscure the whole board, so it's definitely a good investment as far as work and health are concerned.

5. Unknown and hidden traces

As with digital PCBs, the challenging part is usually the remaining 3-5% of the analog PCBs, more so with obsolete or customized ICs that do not have pinouts or reference datasheets. Thankfully, the five points mentioned in [page 171](#) using the common sense approach are still applicable here:

- Refer to datasheet for hint of usage
- Focus on un-validated pins
- Obey electrical rules
- Refer to the PCB's functional block diagram
- Refer to system interconnection diagram

Points 1, 4 and 5 are subject to availability of information. Of course, if all else fail, there's still one¹⁶⁶ useful trick...

¹⁶⁶ Well, maybe not the only one. There's a last-ditch option which I'll discuss at the end of this chapter, but it's not free and you should only consider it if you have spare cash and are desperate enough to want to reverse-engineer your PCB.

ONE USEFUL TRICK

After you've reverse-engineered a number of PCBs, you might begin to wonder if there's a way to locate the connections between component pins faster. Yup, and once I tell you the answer you'd probably say, "Now why didn't I think of that?" Ready? The solution is aluminum foil.¹⁶⁷ What? Yep, you heard it right, mate.

Some of my readers might already got the hint, but there are others who may still need a little more elaborating to turn on the light bulb. No worries—as it is, I'm going to show you two ways you can speed up your search of difficult-to-find electrical connections, especially the elusive ones.

The Magic Carpet Method

This method folds an aluminum foil into a multi-layer rectangular piece of conductive medium, and is then connected to one of the multimeter probe (black) using a cable with crocodile clip ends, as shown in the left figure below. The multimeter must be set to the diode function mode.

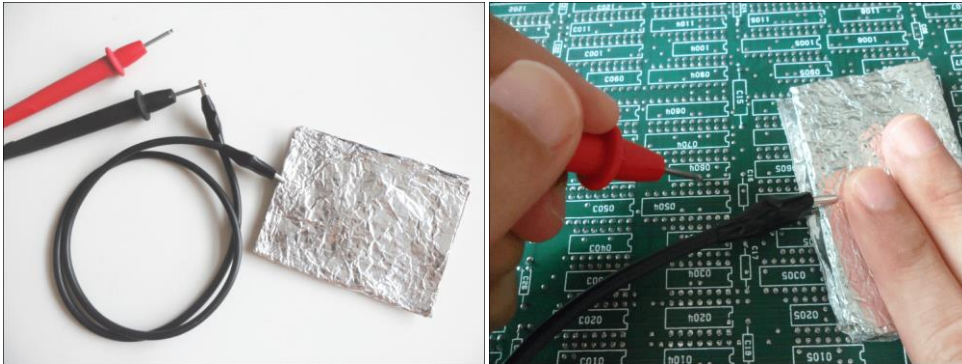


Figure 4.31 The fish net method.

The free probe (red) is used to probe on the component pin of interest, while the folded aluminum medium is pressed on the surrounding areas in systematic, overlapping sequence until a beep is heard, signifying that a low impedance (<100 ohms) or short is detected. You should check the reading each time a continuous beep comes on to make sure it's a valid short. Once it's confirmed, you know that the connecting point is within the area covered by the conductive medium. This way, you could quickly locate by proximity and then revert back to normal probing to pinpoint the exact spot.

The size of the conductive medium is a personal choice but it should not be too big that it leaves a large area for you to probe. Also, if you see that the aluminum foil starts to flake, either turn over to use the other side or change to a fresh new one. The reason is explained in a later section.

¹⁶⁷ Aluminum foils are common household items and can even be found in PCB repair shops, where repair technicians often employ them as protective covers when removing surface mount components using a heat gun or blower, to shield surrounding components—especially plastic materials like connectors and DIN headers—from melting under the intense heat.

The Gold Finger Method

Well—it's not gold actually but aluminum—but I thought gold finger sounded better than an aluminum one (or even silver, for that matter!). The principle is the same as the magic carpet method except this time you wrap the aluminum foil around your index finger, and with one end of the cable clipped to the base as shown in the left figure below:

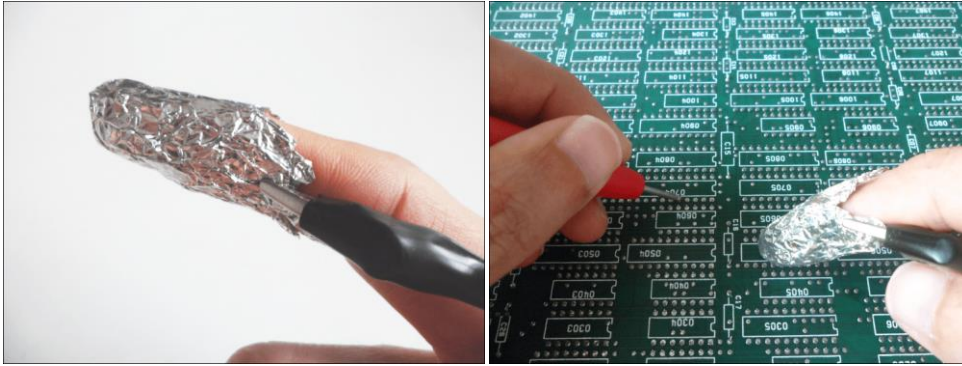


Figure 4.32 The gold finger method.

While the gold finger method does not cover as much ground as the magic carpet, it narrows down the target area once a short is detected. The only flip side is you may inadvertently prick your finger if you happen to press too hard or on a sharp solder point. So make sure you wrap at least 2-3 layers around your index finger!

A WORD OF CAUTION

There are drawbacks to using the aluminum foil that I need to highlight, in case you become excessively dependent upon it to find every single connection on the PCB:

1. Repeated pressing of the aluminum foil against the PCB's solder side, whether using the magic carpet or gold finger method, will render it flaky due to abrasion by the sharp pointed soldered pins of the components. Small fragments will eventually break off from the main body and either lodge directly onto or in between pins on the next contact. Some of these fragments can be so minute and powdery it could well form subtle electrical bridges that short across pins that are within close proximities, and might mislead you to a false electrical connection.

It is therefore important that the PCB is stripped clean of any conformal coating before applying this technique, else these aluminum particles may stick to the board surface and cause possible fault conditions that are hard to solve. Also, you should use an anti-static brush to remove any possible aluminum residues before re-applying conformal coating.

2. The aluminum foil method does not pinpoint the exact pin but only provides a rough estimate of its position in the area where a short is detected, and will still require you to revert back to the probing method to find the correct pin. This can be time consuming compared to simply probing using the various strategies mentioned earlier.

It is always tempting to go brute-force and use what looks like an easy and straightforward method for such as a demanding task as reverse-engineering. But it bears to remember that too much of a good thing may not necessarily be beneficial in the long run, so use this trick in moderation and only when you need to.

DOING THE UNTHINKABLE

If you're in a do-or-die situation where you really need to get a PCB reverse-engineered, for whatever reasons, there is a last-ditch solution which you can consider—but it comes with a price tag, of course. If you've read [Chapter 1](#), you'd know that there's a kind of equipment that can do reverse-engineering faster than any human possibly could. That's right, I'm referring to a flying probe test system! Now, don't get too worked up... I'm not suggesting that you go out and get one of these machines, because they're definitely not cheap, and then there's the training and learning curve too.

I know there are companies out there with these machines and depending on their business model and setup, may provide the following services at a reasonable price:

- PCB scanning and digitizing to obtain X-Y coordinates
- Netlist learning and re-generation of PCB connectivity
- Export netlist data to popular CAD or EDIF formats
- Recreate schematic diagram using target CAD software

Depending on your budget and deadline, you can opt for a partial or complete package of the above services. The onus is on you to weigh expenses against expediency to be sure it does not eat into your profit margins, or sacrifice hard work at the expense of real, satisfying engineering work.¹⁶⁸

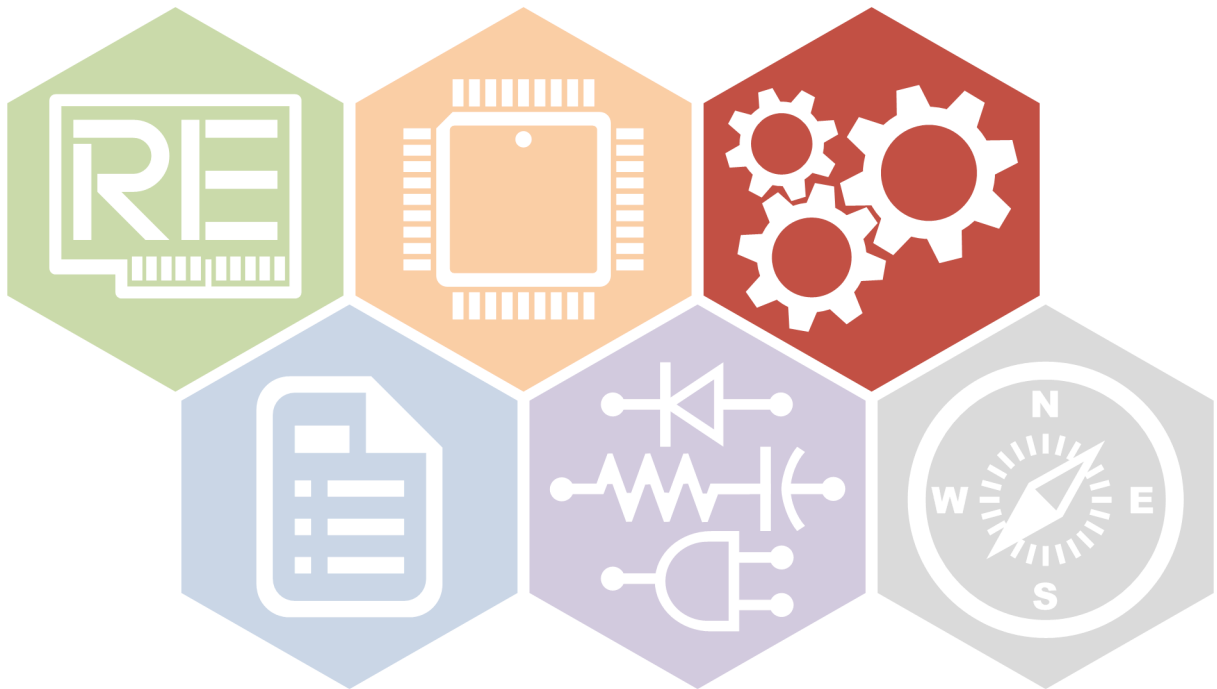
And there you have it—what reverse engineering by hand is all about—or is it?

¹⁶⁸ Unfortunately, the mantra of young electronics engineers today is: Work smart, not hard. With such abundant information at your finger tip in this internet age, it has bred a new breed of engineers that are good in knowledge but lacking in depth through hard-earned experiences.

5

Advanced Topics

**Practice makes perfect,
but sometimes knowing a better way
to do things helps too...**



CONTENT

Layering Technique–203

What is Layering Anyway?–204 When to Use Layering–205

How Layering Works in Visio–206

Adding a Layer–206 Assigning a Shape to a Layer–207 Activating One or More Layers–207

Renaming a Layer–208 Deleting a Layer–208 Show or Hide a Layer–209

Locking a Layer–209 Snap and Glue–209

A Layering Example–210

Summary on Layering–218

Simplicity in Complexity–219

Smartshapes–219 Shape Data–220 Shape Graphics–224

Customizing Your Own Data Graphics–229 ShapeSheets–230

Building a Multi-Field Shape–239 A Multi-Shape (DIP-Switch) Example–250

Adding a Shape Context Menu–255

Summary on Shapes–256

Generating Bill of Materials–257

The Matrix–263

JEDEC and Fuse Map–264 JEDEC De-Compiler–267

Summary–270

5

ADVANCED TOPICS

"Only one who devotes himself to a cause with his whole strength and soul can be a true master. For this reason mastery demands all of a person."

Albert Einstein, 1879-1955

If you've devoted time and effort to work through the pages up till this point, and in the process had some measure of success in your reverse engineering attempts, let me congratulate you for achieving a significant milestone. You have earned your badge of distinction and my respect. You have progress from an initiate to an apprentice. Only one thing stands in your way of mastering this craft—your devotion, or should I say the lack of it.

It's easy to get something started, to try something new; there's excitement and enthusiasm that comes from the exploration of new frontiers and possibilities. But once the novelty wears off and familiarity sets in, interest will dwindle and the acquired skill neglected through disuse. That's why I've decided to include this chapter, to share with you some of the advanced techniques—which will otherwise require time to learn and discover from your own experiences and mistakes—to motivate you further. After all, electronics is a live subject; it does not stay still but keeps evolving and advancing with new and exciting inventions and breakthroughs. Indeed, nobody knows or can predict what kind of new PCB technologies tomorrow will bring.

With this in mind, let's begin...

LAYERING TECHNIQUE

Layering is nothing new in the field of graphics editing software, though it was formerly found only in high-end packages like the famous Photoshop from Adobe Systems, and the de facto AutoCAD from computer-aided design industrial leader Autodesk Inc. As the Open Software community expands and matures, free graphics editing software began to take on these heavy weights by duplicating their many similar and powerful features. Suddenly, layering technique has become one of the basic, must-have capabilities in any respectable graphics editing software to be credible or even usable—raster or vector-based.

If you've never worked with layers before, you're not only limiting yourself to the options and possibilities that are available to bring your engineering illustrations to the next level, you're also doing yourself a big disfavor. Confining to a single flat sheet or layer can be very unforgiving especially with

pixel-based graphics, because a careless mistake can permanently alter and affect the final artwork. Of course, you can undo many times over, or save your precious work frequently—but why subject yourself to unnecessary trouble or torture when there's a better alternative to work with?¹⁶⁹

What is Layering Anyway?

Imagine your drawing page as the base sheet or background on which you can overlay or stack more than one sheet of transparent film, each containing different objects or information that, when combined together and pressed into the background, form the complete drawing. Take for example a PCB layout diagram which comprises mainly ICs, resistors, capacitors and diodes; you can then use the physical frame of the PCB as the background, and segregate the components into different layers, one for ICs, one for resistors, etc. as shown below:

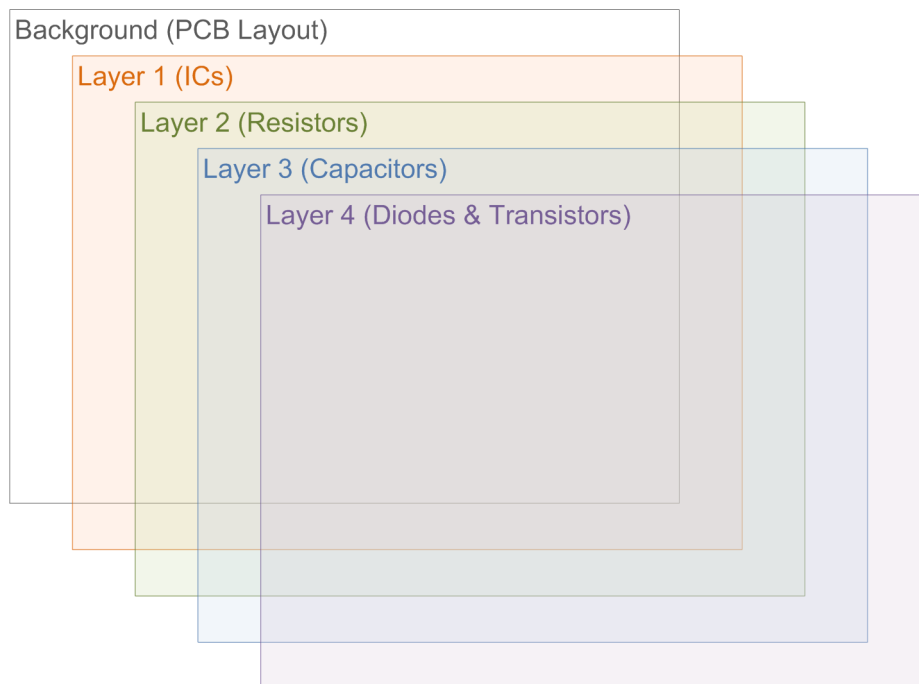


Figure 5.1 A PCB layout diagram with component layering applied.

It may take a while to grasp what I just said but once you understand the concept, you'll realize the power of the layering technique to achieve a new, exciting level of experience in engineering drawing.

¹⁶⁹ By now you should know that Visio is vector-based, which means each drawing entity is manipulated as an object with independent coordinates and properties, which translates to better flexibility and portability when effecting changes to a single object or a group of objects.

When to Use Layering?

It's not a question of why but when you should use layering. They say you can't teach an old dog new tricks but thankfully electronics engineers are a different breed altogether, because we are known to be a resilient lot that constantly look for better (in fact, great) solutions to get our work done, even if we have to break a bone or two in the process¹⁷⁰ (anybody still with me?).

So when do we need to use layering?

1. Most often, I use layering on PCB layouts more than on schematic diagrams. But if the layout is just a simple board like the ISA SCSI host adapter which we did in [Chapter 3](#), then there is no reason to do layering. For complex boards and those with components that are tightly packed like sardines, leaving no room for reference designations, you should consider this option.¹⁷¹
2. Sometimes, you may want to put comments or notes on a schematic diagram while doing circuit analysis or troubleshooting, it's good to put these scribbles on a separate layer so you can turn it off when you want to just print the schematics alone.
3. It is not uncommon to come across PCBs with different revisions and configurations. These PCBs usually use the same bare board design containing extra component footprints to accommodate future growth extension or anticipate obsolescence. Layering technique allows you to handle these variants within the same layout diagram by organizing them using different layers, so that you can choose to turn on the layer with the PCB revision you want to view or print.¹⁷²
4. An added flexibility of layering is the ability to create different layers of the same layout with highlighting emphasis on different portions of the PCB. This is useful when you want to correlate the components to sub-circuits on a schematic diagram, so that at a glance you know where they are located on the PCB to accelerate troubleshooting.

There are certainly many more uses you can probably think of that layering can do. I hope by now your creative juices are flowing and you are raving to want to learn all you can about this feature that comes with Visio. You might even be glad that you've picked Visio as your choice for engineering drawing—and trust me, layering in Visio is a piece of cake—a layered one that is!



¹⁷⁰ Literally! Once I was working in another company's premise and had to help move some heavy power supply equipment. In the process my lumbar vertebrae's knee extensors (L3 section) went out of place, causing me three painful sleepless night. I couldn't even turn to my side while lying down, and had to walked with great difficulty and effort. Thankfully, a colleague recommended me to a Chinese medical practitioner who's specialized in bone adjustment—one look at me as I limped awkwardly into his clinic and he knew where the problem was! Within ten minutes I was up jumping and squatting with great ease and delight. And it cost me less than 30 bucks to get it fixed!

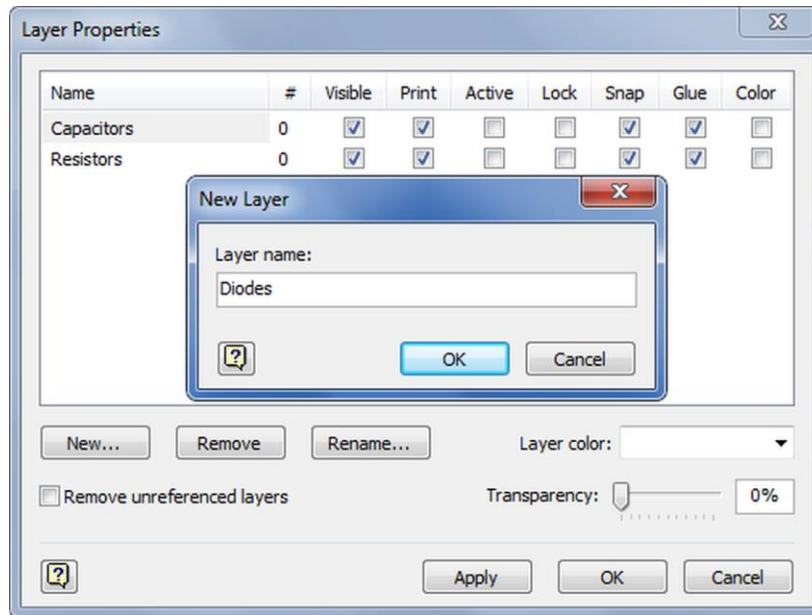
¹⁷¹ I will provide an example of such a PCB that I worked on so you can get a better idea, right after I show you how layering works in Visio.

¹⁷² It works like a kind of version control with the advantage of having to deal with one drawing document instead of managing several, which can be confusing and quite frankly, error prone.

How Layering Works in Visio

In a vector-based software like Visio, where each shape is a distinct entity or object, layers are useful in organizing related shapes on a drawing page. In a sense, a layer is a named category of shapes. By assigning shapes to different layers, you can selectively view, print, color, and lock different categories of shapes, as well as control whether you can snap to or glue to shapes on a layer. And because Visio supports multiple page drawing, each page in a drawing can have its own set of layers.

Adding a Layer



To create or add a layer:

1. On the **View** menu, click **Layer Properties**, and then click **New**.
2. Type a name for the layer, and then click **OK**.
3. In the **Layer Properties** dialog box, select the check box in each column for properties that you want the layer to have, if they are not already checked.

Important to note:

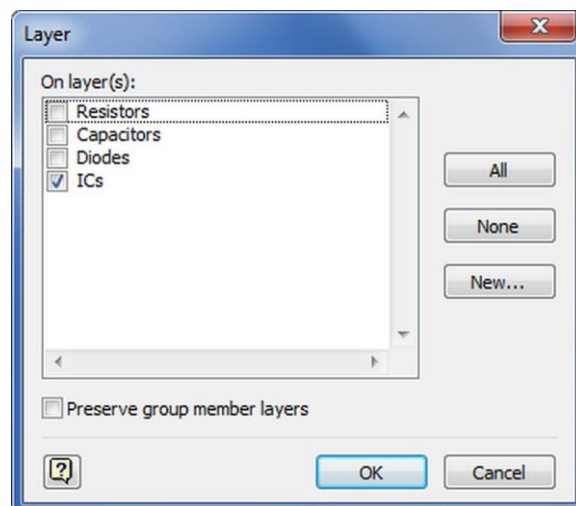
- When you create a new layer, it is added only to the **current page**, not to all pages in the file.
- Similarly, when you create a new page, that new page does not inherit layers from the previous page. You must define the layers you want the new page to have.
- When you copy a shape with a layer assignment from one page to another, either within the same drawing or from one drawing to another, the layer is added to the new page. If the page already has a layer with the same name, the shape is added to the existing layer.

Assigning a Shape to a Layer

A shape can be assigned to a single layer, multiple layers or none. For shapes that are already assigned to layers, when you move or copy them to a page, the corresponding layers are automatically added to that page.

To assign a shape to a layer:

1. Select a shape.
2. On the **Format** menu, click **Layer**.
3. Click the layer or layers to which you want to assign the shape to.



Activating One or More Layers

Making a layer active is a quick way to assign shapes to that layer as you add them to the page. If a shape is not already assigned to a layer, the shape is automatically assigned to the active layer when you add it. Once you activate a layer, all the shapes you add from then on will be assigned to that layer.¹⁷³

To activate a layer or multiple layers:

1. On the **View** menu, click **Layer Properties**.
2. For each layer you want to make active, select the check box in the **Active** column.

Name	#	Visible	Print	Active	Lock	Snap	Glue	Color
Capacitors	0	☒	☒	☐	☐	☒	☒	☐
Diodes	0	☒	☒	☐	☐	☒	☒	☐
ICs	0	☒	☒	☒	☐	☒	☒	☐
Resistors	0	☒	☒	☐	☐	☒	☒	☐

Note:

- The layer or layers are active only for the current page.
- You cannot activate a layer that is locked to prevent editing.

¹⁷³ You can designate more than one active layer. Shapes you add to the page are automatically assigned to all of the active layers.

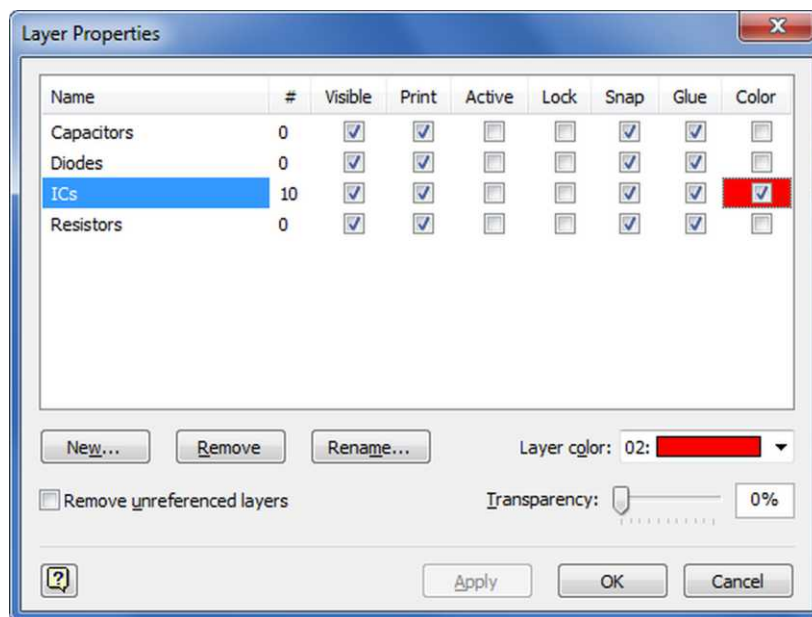
Renaming a Layer

1. On the **View** menu, click **Layer Properties**.
2. Select the layer you want to rename, and then click **Rename**.
3. Type a new name, and then click OK twice.

Note: The layer is renamed on the current page only. The shapes on the layer are not removed or changed.

Deleting a Layer

Before deleting a layer, it is important to check if any shapes you want are assigned to that layer. You can do a quick check by assigning a color to that layer:



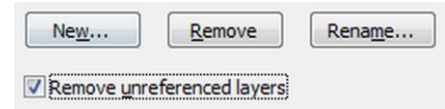
1. On the **View** menu, click **Layer Properties**.
2. Check the number column '#' to see if any shapes are assigned to the layer. In this case there are 10 shapes assigned to the **ICs** layer. To determine which are the 10 shapes, click the **Color** column and select say **red** in the **Layer color** drop down menu, and click **Apply**. Shapes assigned to the **ICs** layer will turn **red** for easy identification.¹⁷⁴
3. Select these shapes. On the **Format** menu, click **Layer**. Now click the layer you want to re-assign the shapes to, then click OK.

¹⁷⁴ Assigning colors to layers do not change the original colors of the shapes on those layers. Once you uncheck the **Color** column for a layer, the shapes on that layer will revert back to their normal colors again.

Once you've taken care of the shapes you want to retain, you can now proceed to delete that layer as follows:

1. On the [View](#) menu, click [Layer Properties](#).
2. In the [Layer Properties](#) dialog box, select the layer you want to delete, and then click [Remove](#).

Note: To delete all unused layers, in the [Layer Properties](#) dialog box, check [Remove unreferenced layers](#), then click OK.



Show or Hide a Layer

The power of layering technique lies in its ability to turn on (show) or turn off (hide) specific layer(s) both for display or printing.

For display visibility:

1. On the [View](#) menu, click [Layer Properties](#).
2. Under [Visible](#), click to select or clear the check box for the layer to show or hide it.

For print visibility:

1. On the [View](#) menu, click [Layer Properties](#).
2. Under [Print](#), click to select or clear the check box for the layer to print or omit printing it.

Locking a Layer

1. On the [View](#) menu, click [Layer Properties](#).
2. Under [Lock](#), click to select or clear the check box for the layer to be locked or unlocked.

Snap and Glue

You can enable or disable the snap and glue for individual layers by following the same procedures as mentioned above.

Now that we have a better understanding of how layering works in Visio, it's time to take a look at an example...

A Layering Example

Consider the following mixed-signal PCB containing analog and digital components with a mix of thru-hole and surface mounted packages. For the most part, it can be segregated into two halves, the top half comprising mainly digital ICs and their decoupling capacitors along with a few pull-up SMD resistors, and the bottom half which is mostly analog ICs and their decoupling capacitors together with supporting discretes to form various signal conditioning and driving amplifier circuits.

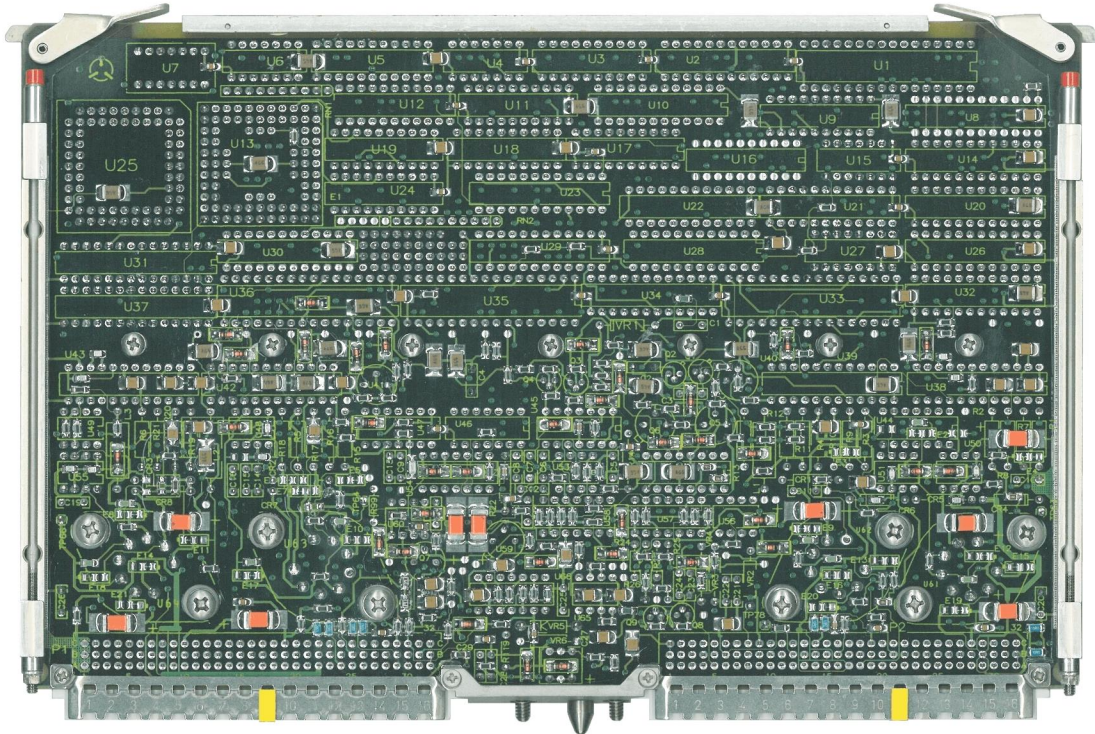


Figure 5.2 A typical mixed-signal PCB (solder side view).

You can clearly see the contrast between these two halves—the top half is well spaced out with only the decoupling capacitors left unmarked, while the bottom half is quite congested and the SMD components are not labeled at all.

Now, if you draw and place all the component footprints on the default drawing page consisting of only a single layer, that's fine—if you do not intend to furnish the unmarked components with self-assigned reference designators¹⁷⁵—but you're going to have a problem identifying¹⁷⁵ them when you start to draw the schematic diagram. And if you decide to name the unmarked components, you are still faced with the challenge of space constraint to place those reference designators on just a single layer without overlapping and obscuring them, and worst still, end up with a messy layout that does not serve any useful purpose!

¹⁷⁵ Like they say, "How on earth are you going to tell who's who in the zoo?"

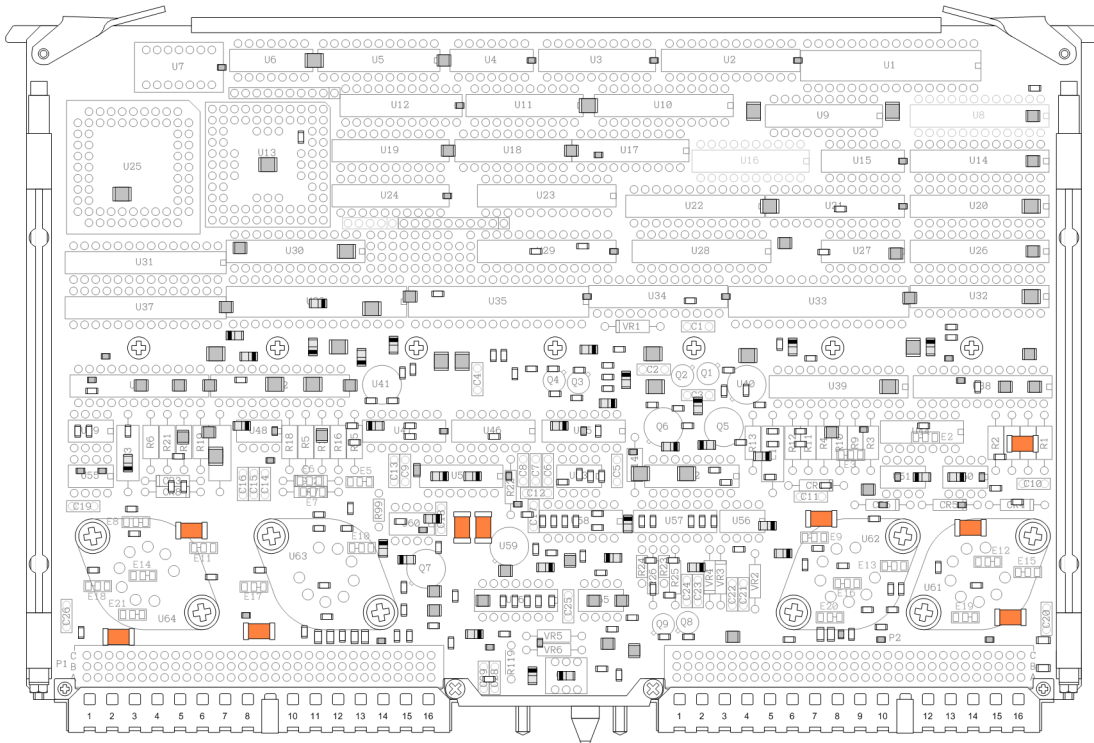


Figure 5.3 Layout diagram of the mixed-signal PCB.¹⁷⁶

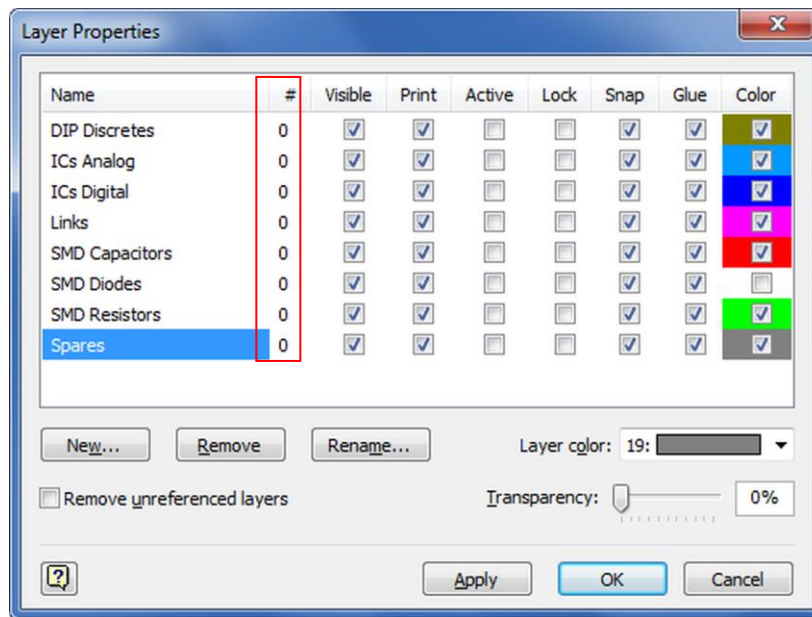
That's where layering comes to the rescue. Using this technique, however, requires careful planning or you may find yourself with lots of clean-up work to do later on. Firstly, we need to survey the PCB to get a good demographic make-up of its components. For this PCB, we can group the components into the following sets:

- Analog integrated circuits (ICs)
- Digital ICs
- Through-hole discrete components
- Surface mount (SMD) capacitors
- SMD resistors
- SMD diodes
- Links or jumpers
- Spare or reserved component footprints

Once we identify and group the components into related categories, we can start to create or add new layers by selecting the [View](#) menu, choose [Layer Properties](#), then click [New](#) and start adding the layers accordingly. There are altogether eight layers and we can further distinguish these layers by assigning different colors to each of the layers.

¹⁷⁶ You can choose to draw all the components and place them on the default drawing page first, and then assign them to their respective layers, or create the layers first, then activate each layer in turn as you draw and place the components. For simplicity's sake, let's assume we took up the first option.

Here's how the completed [Layer Properties](#) dialog box looks like:



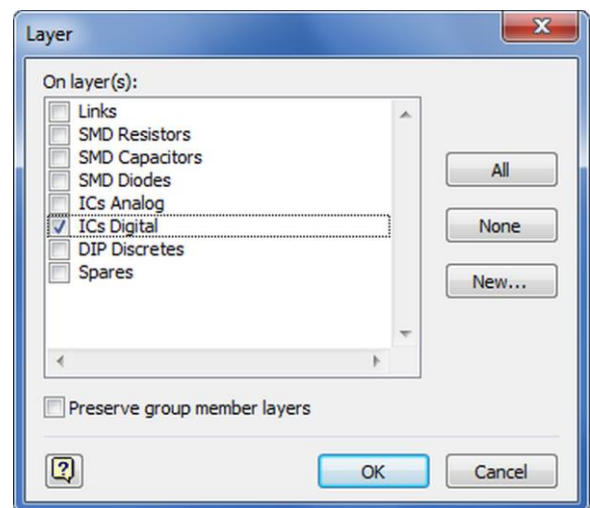
You can tell that there are no shapes or objects assigned to any of the layers yet by looking at the number column '#' (red box area). Also, it is important that none of the layers are active at this time. After clicking OK, we can start adding the components to their respective layers.

Let's say we want to add all the digital ICs to the [ICs Digital](#) layer. To do so, you can:

- Select one IC at a time,
- A range of ICs, or
- All the ICs at one go,

then go to [Format](#) → [Layers...](#) and tick the [ICs Digital](#) layer name in the [Layer](#) dialog box (see figure).

Note: I would advise against being overly ambitious to select all the digital ICs at one go, in case you slip up and accidentally click on a non-digital IC object or shape, or unwittingly click on an empty area and de-select all the ICs, in which case you'll have to start all over again.



As you incrementally add digital ICs to the [ICs Digital](#) layer, those ICs which become members of that layer will take on the layer's color—in our case, it's [blue](#). Should you make a mistake and assigned a non-digital IC component to the [ICs Digital](#) layer, simply select that component, then [Format](#) → [Layers...](#) and untick the [ICs Digital](#) layer name in the [Layer](#) dialog box will do.

Once you're done assigning all the components to their respective layers correctly, the final PCB layout will reflect the colors of the various layers on their related components, as shown below:

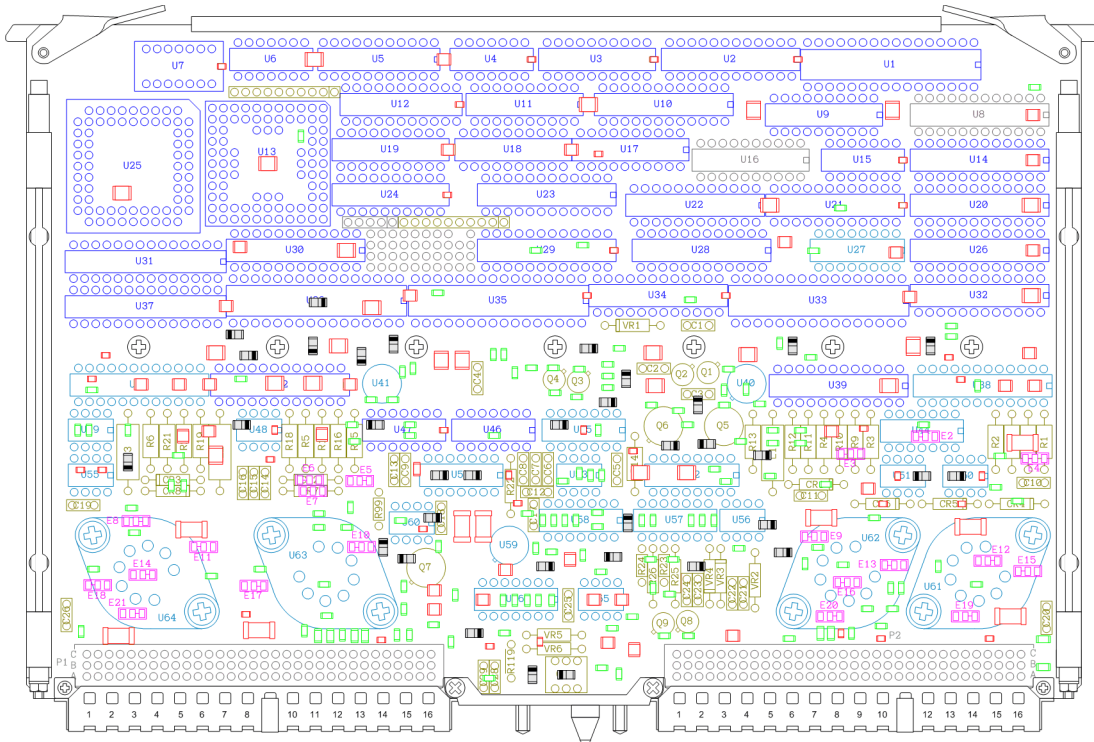


Figure 5.4 Layout diagram of the PCB after layer assignment.

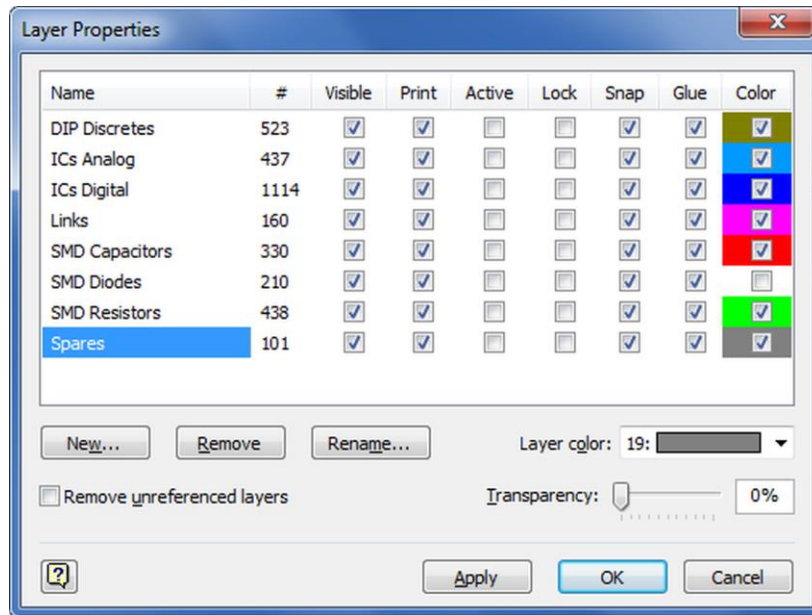
If you have taken the trouble to color and shade your components as depicted in Figure 5.2, you may be wondering or perhaps even gotten upset that the colors and shades have all but disappeared after layer assignment is completed. Not to worry—your hard work is not gone to waste. To see the original colors and shades, simply bring up the [Layer Properties](#) dialog box and uncheck the [Color](#) columns and the PCB layout diagram will be restored back to its original glory.¹⁷⁷

The fact is, you do not need to assign colors to the layers at all, especially if you're not going to print out the artwork in color. But having colors make it easier to differentiate the component types and to check if any component has been inadvertently assigned to the wrong layer, and the good news is the original colors and shades are preserved intact in case you want it back. That's the beauty and power of Visio's layering function.

Personally, I prefer to complete the whole PCB layout and then create the layers and do the component assignments, instead of the other way around. It's much quicker and less error prone.

¹⁷⁷ You can also restore an individual component or a group of components to their original colors and shades by selecting them, then go to [Format](#) → [Layers...](#) and untick the layer they're assigned to, but this will remove them from the layer and you will have to re-assign them back again, if necessary.

The [Layer Properties](#) dialog box after the layer assignment is completed:



Notice the number column (#) displays a list of random numbers against the layer names. These are the total number of shapes or objects found on each of the layers, but if you're sharp enough, you'd see that something is amiss here—the numbers are much bigger than the actual component count! Why is that so? Well, it has to do with how the components were created in the first place, and I will delve a little into this topic in a later section, when we want to get Visio to generate a bill of materials from the PCB layout diagram.

Now that we have assigned all the components to their related layers, we can start to name those unmarked SMD parts. Two options are available:

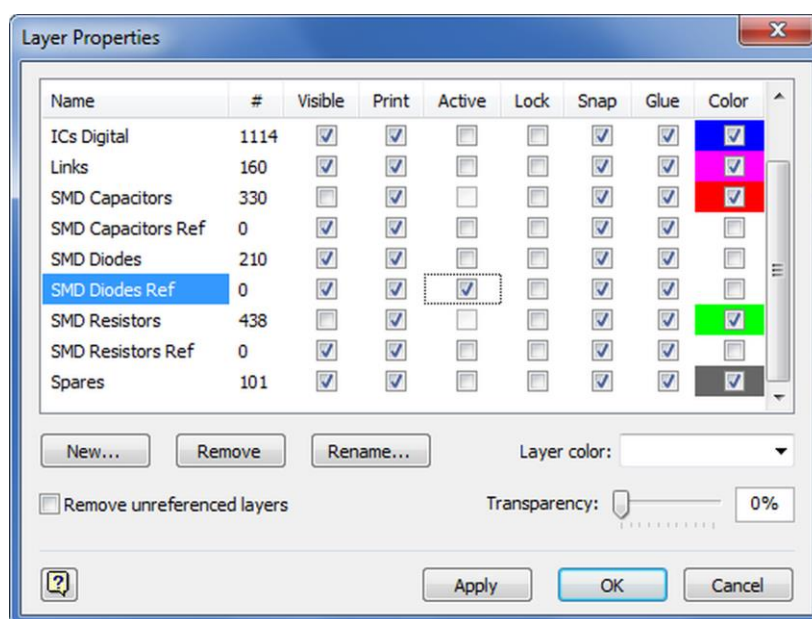
- Place the reference designators on the same layer as their components. This is straightforward but it has a problem—when all the layers are visible, overlapping of text will occur.
- Create separate layers for reference designators so you can turn them on and off individually to achieve the best possible visual effect and printout.

We will go for the second option and create three additional layers for the SMD parts:

- SMD Capacitors Ref
- SMD Diodes Ref
- SMD Resistors Ref

Once you're done, make one of the Ref layers (e.g. diodes) active while leaving the other two inactive. It's OK to leave all three visible since there are no text objects on any of them yet. Next, make the SMD capacitors and resistors layer invisible by un-checking their [Visible](#) column.

The [Layer Properties](#) dialog box should now look like that:



Click OK. The PCB layout diagram will suddenly look more spacious without the [SMD Capacitors](#) and [SMD Resistors](#) layers:

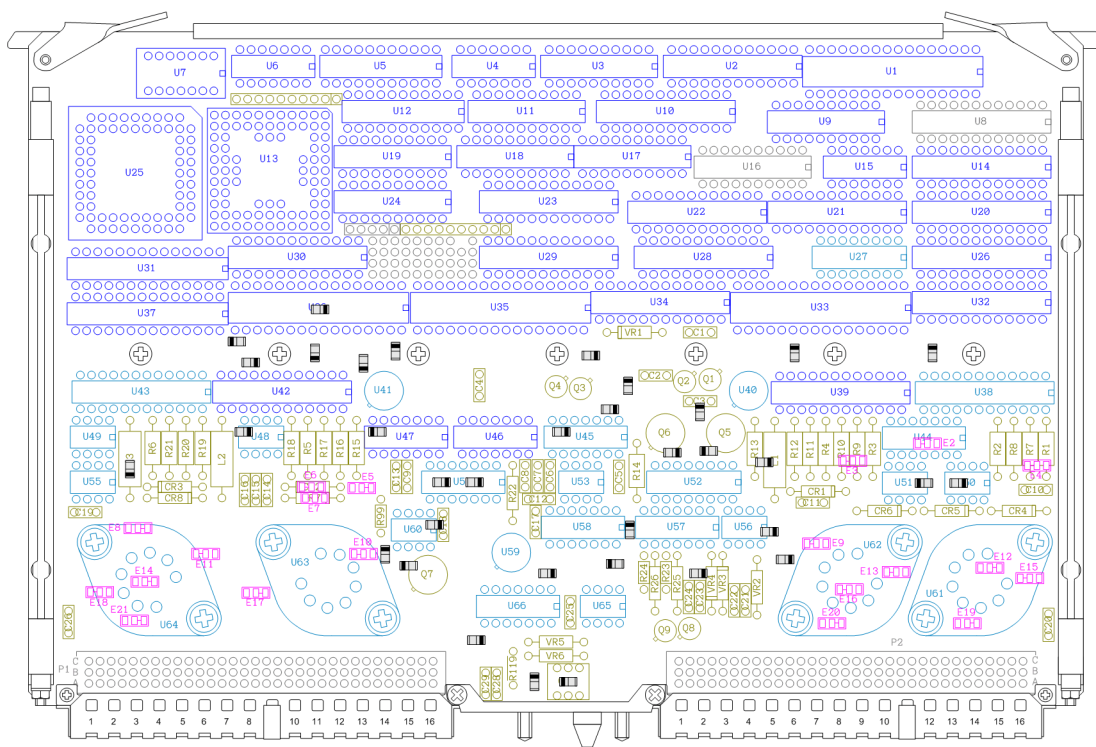
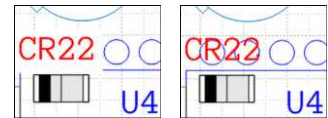


Figure 5.5 PCB layout without capacitor and resistor layers.

We can now proceed to add text objects to the active [SMD Diodes Ref](#) layer. Since these SMD diodes do not have any reference designators of their own, we'll have to individually assign them ourselves. We need to ask the following questions:

- What kind of prefix to use? Fortunately, there are some through-hole diodes which we can take reference from—in this case, it's 'CR'.
- What is the starting number to give? Since we're going to use the same prefix as the through-hole diodes, we should find the largest number in use—in this case, it's 'CR8' so the next running number prefix should be 'CR9' and so on.
- How should we assign the reference designators? Normally, the number prefixes of components on a PCB run either horizontally or vertically. For this PCB, the ICs are laid out with their number prefixes running horizontally from right to left (solder side), but the discrete components do not follow this arrangement. No matter. We will stick to the IC's paradigm to be more organized and just in case we miss out a number or two in-between, or come up with duplicates.

Note when adding text: make sure to fill the background color to solid white and use **red** for text color so it stands out when put alongside the diode (left figure), and not become blend in with the surrounding artwork (right). No worries, you can easily change the text color later.



The PCB layout should look like this once we're done:

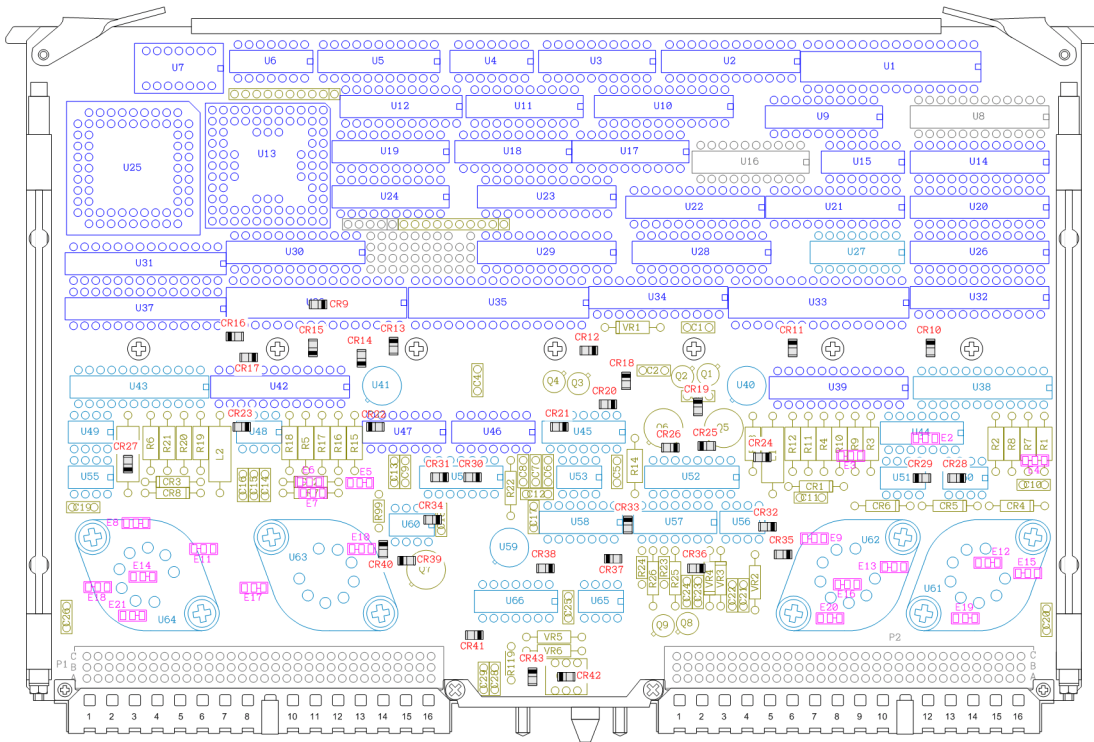


Figure 5.6 PCB layout with SMD diodes assigned reference designators.

Now turn off the **SMD Diodes** and **SMD Diodes Ref** layers i.e. uncheck their **Visible**¹⁷⁸ boxes, and in turn set the **SMD Capacitors** to visible and make the **SMD Capacitors Ref** layer active, then proceed to assign reference designators to them like we did for the SMD diodes.

It's definitely more challenging to assign reference designators for the SMD capacitors (and possibly the SMD resistors later), because of their large numbers and through-hole counterparts that need to be taken into consideration. One trick or method is to assign different prefixes to the decoupling capacitors (**CU**), the big tantalum capacitors (**CF**), and the rest of the normal signal capacitors (**C**).

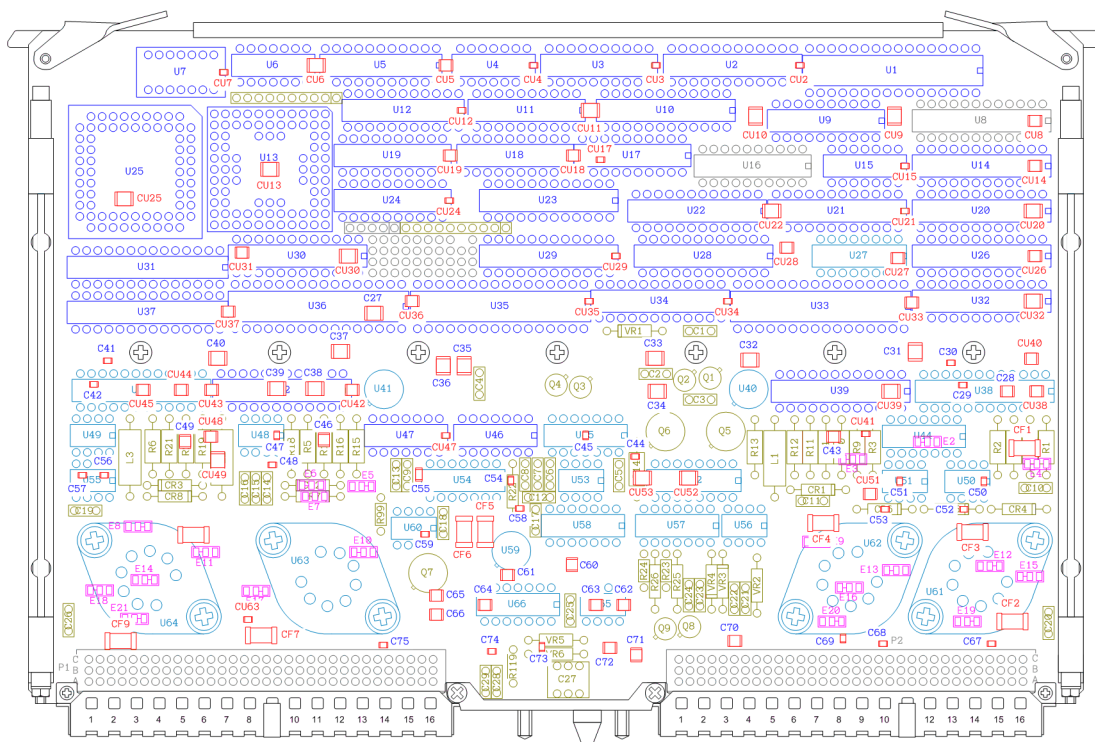


Figure 5.7 PCB layout with SMD capacitors assigned reference designators.

Note that the decoupling capacitors are numbered as closely to their owner ICs as possible, e.g. for U2 it'll be CU2, and so on. To further differentiate between the decoupling capacitors and signal capacitors in case we get confused and make wrong assignments, it's a good idea to use different text colors—such as **red** for the decoupling capacitors and **blue** for the signal capacitors, as shown in the figure above.¹⁷⁹

Similarly, we'll now turn off the **SMD Capacitors** and **SMD Capacitors Ref** layers, set the **SMD Resistors** to visible and make the **SMD Resistors Ref** layer active, and assign reference designators to them.

¹⁷⁸ There is no need to de-activate the **SMD Diodes Ref** layer; once you un-check its **Visible** box, the layer will automatically become inactive as well.

¹⁷⁹ This is the reason we did not assign colors to the **Ref** layers earlier on when we created them—to allow us the freedom and flexibility to use different text colors when the situation warrants it.

The final installment:

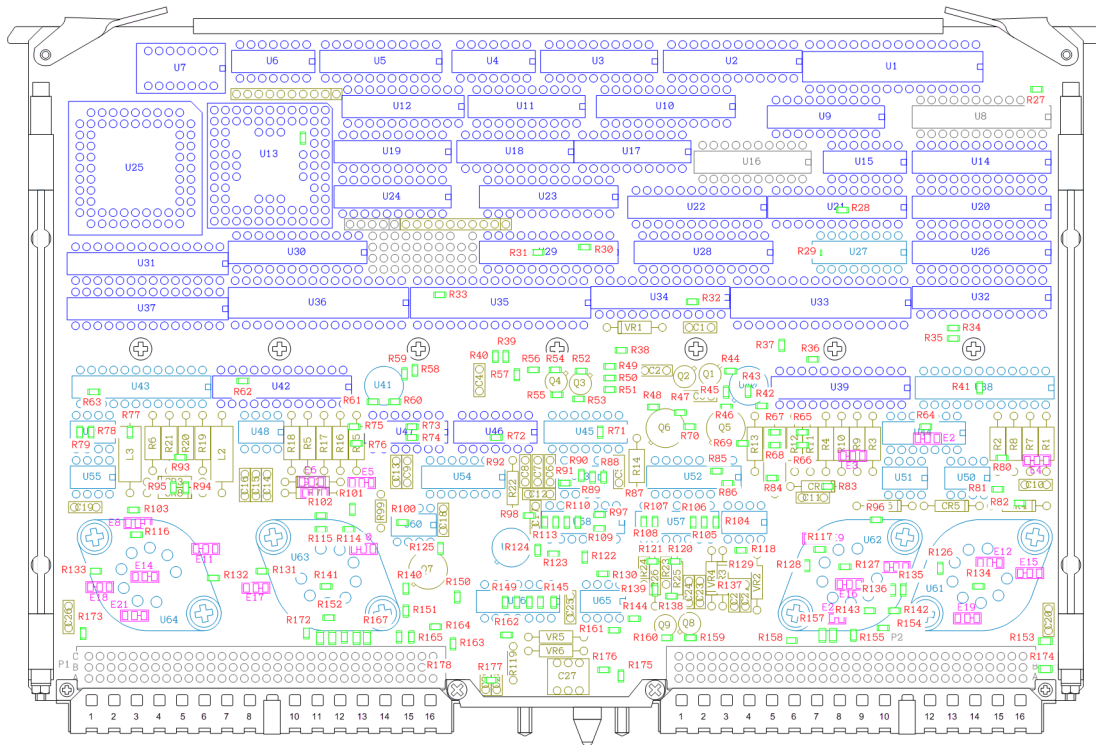
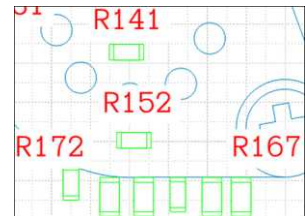


Figure 5.8 PCB layout with SMD resistors assigned reference designators.

Again, sharp-eyed readers will notice that some SMD resistors' reference designators are missing in the diagram above. No prize for guessing why. The answer is space constraint. However, all is not lost as you can infer based on their positions (right figure) that the resistors after R167 leading up to R172 will have running numbers from 168 to 171. This illustrates a point: it is not necessary to put in every reference designators if space does not allow you to, and also when you can infer from the obvious.



Summary on Layering

Hopefully through this example you would have attained a better appreciation of the power of layering in Visio and know how to apply it in your drawings. If you're good in visualization, you'd have realized that layering is not just limited to what has been presented so far. But I'll have to leave it to your imaginations and creativity to think of other possibilities, and move on to something more challenging but important...

SIMPLICITY IN COMPLEXITY

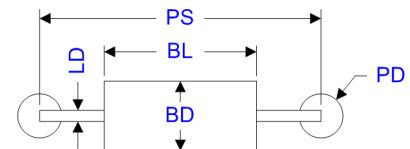
Up till now, we have been using Visio's Drawing Tools (rectangle, ellipse, line and arc) to create basic shapes and group or join them into more complicated objects that serve our purpose for PCB layout and schematic diagram. This is no doubt the fastest way to get started using Visio, but unlike basic graphics editors that treat primitive shapes as simply vectorized entities that can be reconstructed based on X-Y coordinates and geometric parameters, shapes in Visio are far more sophisticated than their deceptively simple looks and ease of use. That's why Visio shapes are also called smartshapes.

Smartshapes

If you have never heard or used smartshapes, then you have not tapped into the true power of Visio. "Just what are smartshapes anyway?" you asked. Smartshapes are the building blocks of Visio, shapes that you drag and drop from the rich variety of stencils included with Visio, even the simple shapes that you create using Visio's Drawing Tools are inherently smartshapes—programmable entities that can exhibit real-world behaviors. In fact, if you understand the principle and inner workings of smartshapes, you can cut down a whole lot of redundant representation of similar shapes with just a single adaptable and customizable shape.¹⁸⁰

Consider the simple axial-lead resistor for a PCB layout symbol. It has a body, two terminal leads, and a pair of solder pads—all of which, parametrically speaking, can vary in different geometrical aspects. For example:

- The size of the resistor's body for 0.125, 0.25, 0.5 and 1-watt models will obviously be different, increasing in bulk with higher power ratings.
- The thickness of the terminal leads will vary with wattage as well, not to mention the length as a result of the solder pads' spacing to accommodate the resistor's physical size.
- Of course, the solder pads will have different diameters to fit the thickness of the terminal leads, plus their spacing which we've just mentioned.



Rating (Watt)	Body length (BL)	Body Ø (BD)	Lead Ø (LD)	Pad Ø (PD)	Pad spacing (PS)
0.125	3.0 ± 0.3	1.8 ± 0.3	0.45 ± 0.05	0.075-inch	0.35-inch
0.250	6.5 ± 0.5	2.5 ± 0.3	0.60 ± 0.05	0.085-inch	0.50-inch
0.500	8.5 ± 0.5	3.2 ± 0.3	0.60 ± 0.05	0.100-inch	0.60-inch
1.000	11.0 ± 1.0	5.0 ± 0.5	0.80 ± 0.05	0.125-inch	0.80-inch

Unless otherwise specified, dimensions are in mm.

¹⁸⁰ Microsoft Visio MVP Chris Roth (aka Visio Guy) said that a smartshape is worth 1000 symbols. It's not too far-fetched to say that a smartshape does have the potential to replace a dozen or so visually similar symbols, but to reach that kind of number will require a lot of creativity and some measure of ingenuity thrown in...

Given the recommended specifications above, you'd probably draw four distinct symbols based on their dimensions using the normal approach discussed in [Chapter 3 on PCB Layout](#). There's nothing wrong with that. As a beginner you should at least give the basic stuff a try and build up your proficiency in using Visio, before jumping into the deep end.

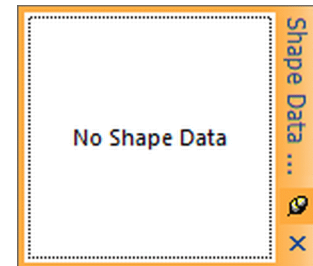
Now comes the crunch. Similarly rated resistors do not necessarily have their solder pads spaced equally apart. A 0.25W resistor may have its solder pads spaced 0.5-inch on one PCB, and 0.6 on another.¹⁸¹ Going by conventional approach, you'll need to create a separate 0.25W resistor layout symbol to suit that particular PCB. And with so many PCB design engineers, each having his or her own style and preferences, you'd probably be creating variant after variant of the same layout symbols and finding it harder and harder to keep track and even to select the right one to use. With smartshapes, your task of managing shapes will become easier, except that you'll need to do the hard work upfront (and thankfully only once) for each kind of shape or symbol.¹⁸²

But before we start creating smartshapes, we need to get acquainted with three components in Visio that makes smartshapes well... smart.

Shape Data

Formerly known as [Custom Properties](#) prior to Visio 2007, the name has now changed to [Shape Data](#). This is probably more of a decision to make things related to shapes in Visio consistent because apart from the name change, everything about it has remained the same.

Shape Data is information or data that's embedded within shapes, and that information is the key that differentiate a smart shape from a dumb one! ¹⁸³ To master smartshapes, the first step you need to know is how to create and manipulate Shape Data. To determine if a shape contains custom data, you can open up the [Shape Data Window](#) from the [View](#) menu, then click on the shape of interest. If no data are present or added to that shape, the Shape Data Window will indicate that there is no shape data.



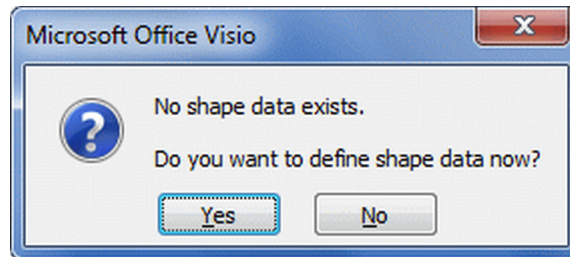
¹⁸¹ Two resistors may not even have their solder pads spaced similarly apart on the same PCB!

¹⁸² Creating smartshapes, on the other hand, may or may not get easier with practice, because each smartshape has its own unique set of requirement and properties to grapple with. But hey, life wouldn't be interesting and worthwhile without real challenges, isn't it?

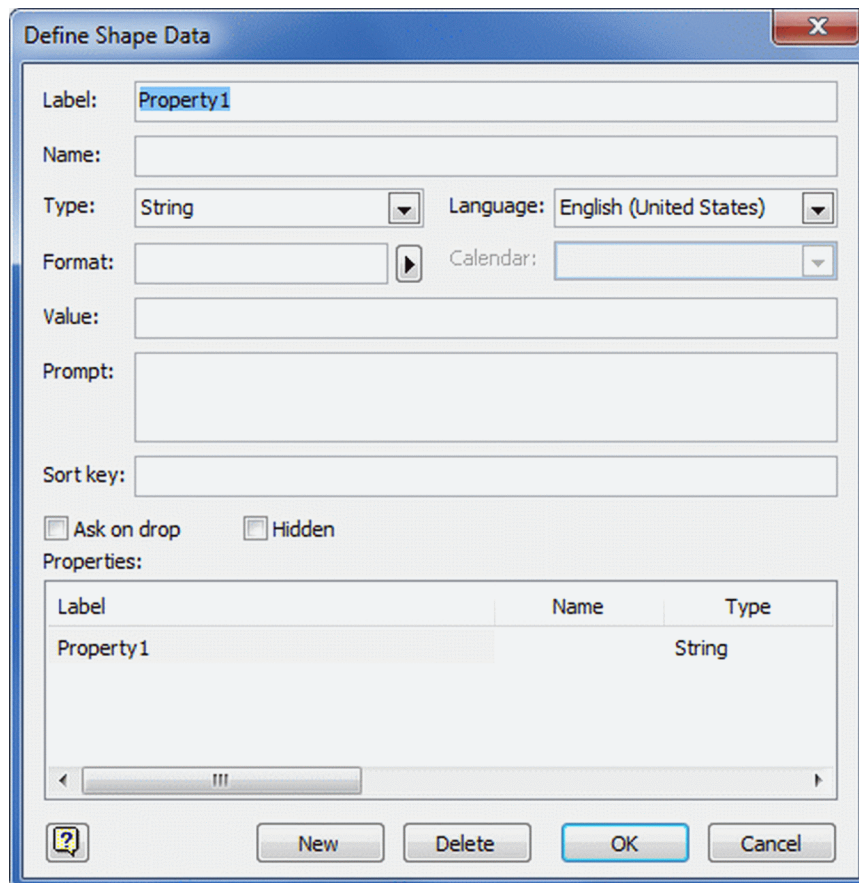
¹⁸³ "Information has the power to change our lives. It can increase the quality of human experience, create new environments to realize our dreams and help us to get beyond problems that trouble our world." —Guido Van Steendam

To add custom data to a shape, say a resistor layout symbol (the one shown on a previous page, which you should be able to draw based on one of the four specifications and group by now):

1. Select the resistor, right click and choose **Data** → **Shape Data...**
2. Since there's no data in this shape, the following dialog box will appear to ask if you want to add data to that selected shape:



3. Click **Yes** and the **Define Shape Data** dialog box will appear:



Notice that there are quite a number of fields available but it is not necessary to fill them all. What you need to take note are the **Label** and **Name** fields—the **Label** field indicates what will be displayed while the **Name** field is of use only if programming is required. For now, we're only

interested with the [Label](#), [Name](#) and [Value](#) fields. Leave the [Type](#) as default [String](#) without any formatting.

4. We will create three data entries for this resistor layout symbol:
- Reference Designator
 - Part Number¹⁸⁴
 - Value

As you fill up the fields, the [Properties](#) table will reflect the data entered. Click [New](#) after filling up each Shape Data page so that a new page will be created for the next data entry.

This is what it'll look like when we're done:

Define Shape Data

Label: VALUE

Name: Res_Val

Type: String Language: English (United States)

Format: Calendar:

Value: NNN

Prompt:

Sort key:

☐ Ask on drop ☐ Hidden

Properties:

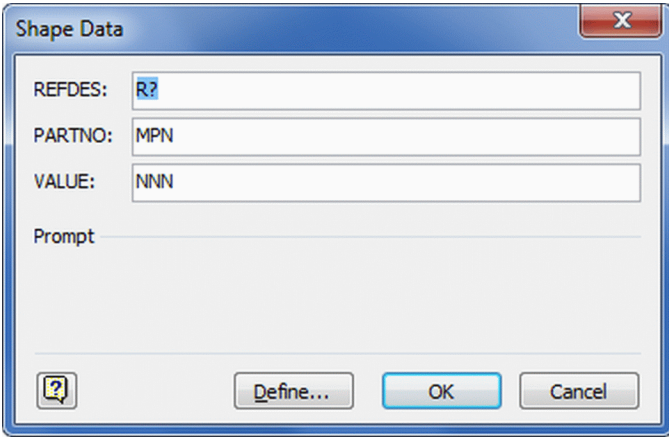
Label	Name	Type	Format	Value
REFDES	Ref_Des	String		R?
PARTNO	Part_No	String		MPN
VALUE	Res_Val	String		NNN

New Delete OK Cancel

For the [Label](#) and [Value](#) fields I usually use UPPERCASE to denote that these are associated with the visible aspects of the Visio shape, while for the [Name](#) field I chose a form of Hungarian-like notation used in computer programming for identifier or variable naming purpose.

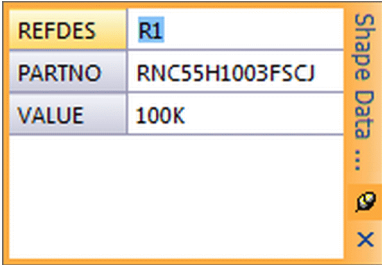
¹⁸⁴ The PARTNO's [Value](#) field is denoted as MPN which stands for Manufacturer's Part Number. You can create another entry named NSN for NATO Stock Number, if desired.

5. After you've entered all three data entries for the resistor layout symbol, click [OK](#). Because this is a new shape data definition, there are no custom shape data yet so immediately a [Shape Data](#) dialog box appears, prompting you to enter the necessary information:



The image shows a 'Shape Data' dialog box with a blue title bar and a close button (X) in the top right corner. Inside the dialog, there are three input fields: 'REFDES:' with the text 'R?', 'PARTNO:' with the text 'MPN', and 'VALUE:' with the text 'NNN'. Below these fields is a 'Prompt' label followed by a large empty text area. At the bottom of the dialog, there is a help icon (question mark in a circle), a 'Define...' button, an 'OK' button, and a 'Cancel' button.

6. Enter the following values: REFDES = [R1](#), PARTNO: [RNC55H1003FSCJ](#), VALUE = [100K](#), then click [OK](#). Notice that the [Shape Data Window](#) now shows the three parameters or data entries for the labels which we've created:



The image shows a 'Shape Data' window with a yellow border. It contains a table with three rows and two columns. The first row has 'REFDES' in the first column and 'R1' in the second column. The second row has 'PARTNO' in the first column and 'RNC55H1003FSCJ' in the second column. The third row has 'VALUE' in the first column and '100K' in the second column. To the right of the table is a vertical label 'Shape Data ...' and a close button (X) at the bottom right.

REFDES	R1
PARTNO	RNC55H1003FSCJ
VALUE	100K

We're done at this point with adding custom data to a shape, but apart from the [Shape Data Window](#), nothing seemed to have changed for that resistor layout symbol. Not to worry, we're getting there soon...

While [Shape Data](#) gives you the ability to embed custom information into a shape, [Data Graphics](#) (or shape graphics) enables you to display that custom information. And that is what we're going to talk about next.

Shape Graphics

Data graphics, or shape graphics, provides four distinct visible graphical representation of embedded shape data that's not only customizable, but able to respond to data variations by changing their visual aspects accordingly, giving an added dimension of intelligence and interactivity to the otherwise boring static shapes:

Text Callouts

Text callouts are actually smartshapes that display textual or numerical information of their host smartshapes' shape data fields' values in which they are associated with.

Data Bars

Data bars too are smartshapes that display graphical indicators of their host smartshapes' shape data values, an alternative to the textual or numerical information by text callouts. Visio supplied indicator graphics include progress bars, speedometers, thermometers, star and people rankings, to name a few.

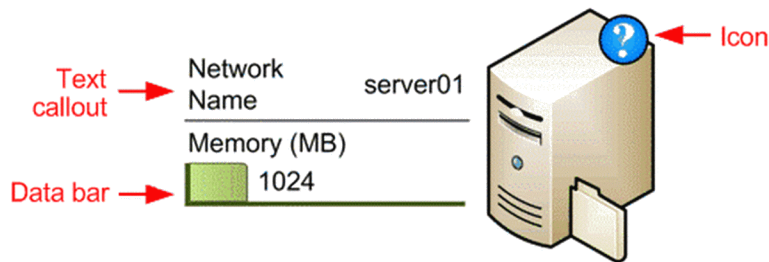
Icon Sets

Icon sets go one step further than data bars by allowing the use of different graphics based on their host smartshapes' shape data values.

Color-by-Value

Finally, the **color-by-value** data graphics create the effects directly on their host smartshapes instead of separately using external representations like the other three above. This can be a very useful feature such as highlighting a component symbol that is obsolete or a customized part.

The first three types of data graphic elements are shown in the network server illustration below to give you a better idea of their usage:



While Visio provides these data graphics for use with smartshapes, it is not necessary to use them all for a particular type of drawing due to the relevance of content to be displayed. Some are more suited for organizational charts, while others bring out important information in a network configuration setting, as can be seen in the examples overleaf.

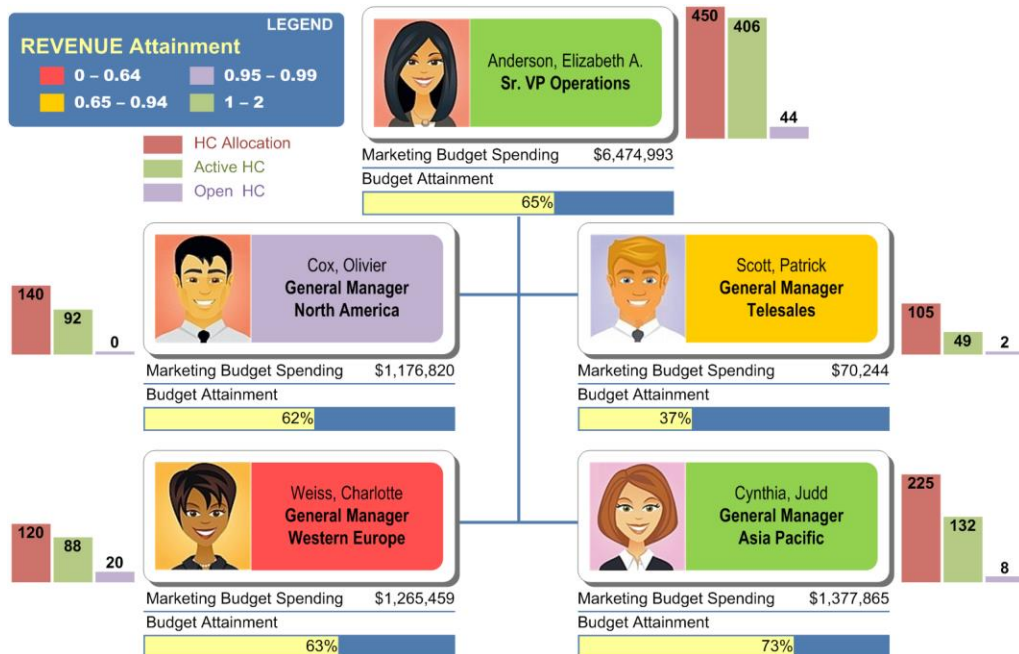


Figure 5-9 An organization sales performance chart with data bars.

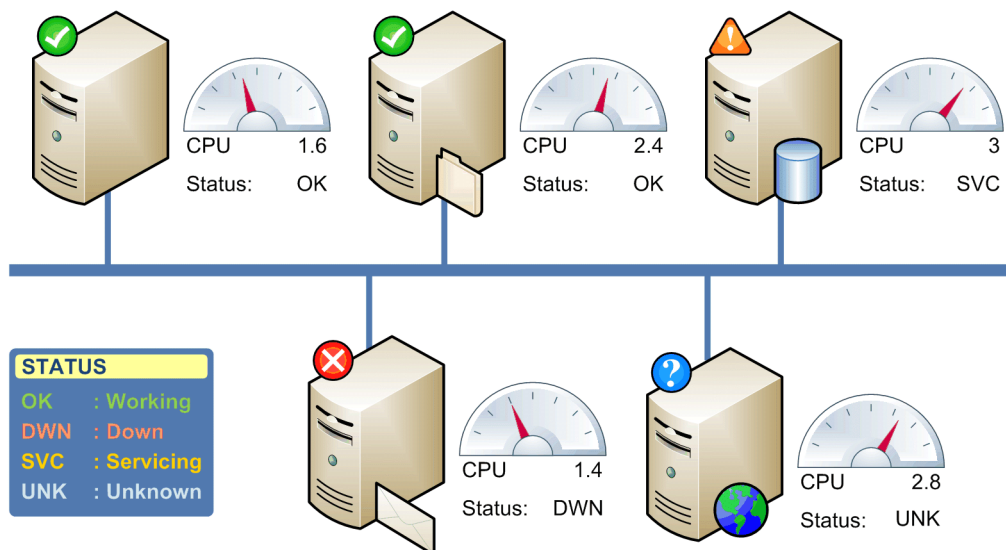
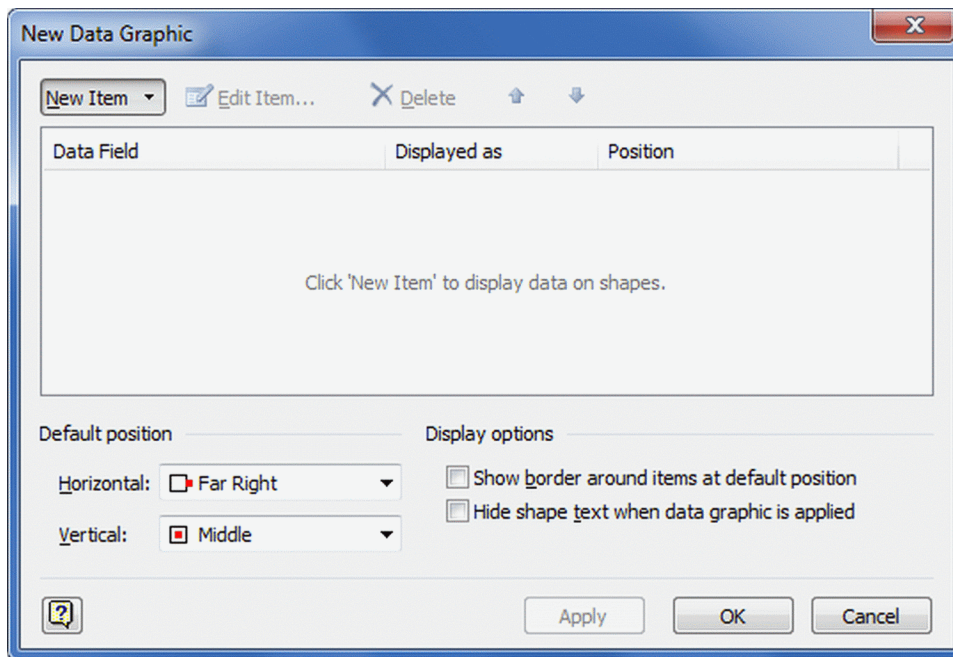
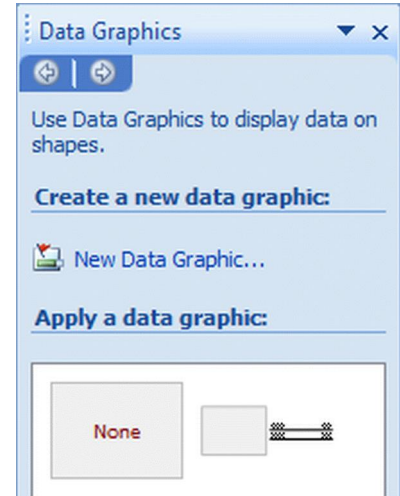


Figure 5-10 A network configuration setting with icon sets.

Colorful and diverse as they are, we will only be making use of the text callout data graphic for our PCB layout symbol, which is adequate for our purpose for now.¹⁸⁵ Going back to the resistor shape which we've created and added shape data earlier, let's add a text callout. There are two ways to apply data graphics to a shape:

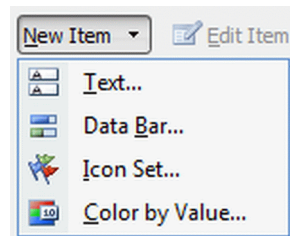
- Go to main menu **Data** → **Display Data on Shapes...** The Data Graphics panel will appear on the default right hand side of the Visio drawing page, with a list of data graphics that you can apply to your selected shape. You can also click on the **New Data Graphic...** link to create a new data graphic from scratch. The New Data Graphic dialog box will appear, and you are ready to associate the shape data to your choice of graphic representation.
- Alternatively, you can right click on the shape to bring up the context menu and then select **Data** → **Edit Data Graphic...** which will directly bring up the New Data Graphic dialog box as shown:



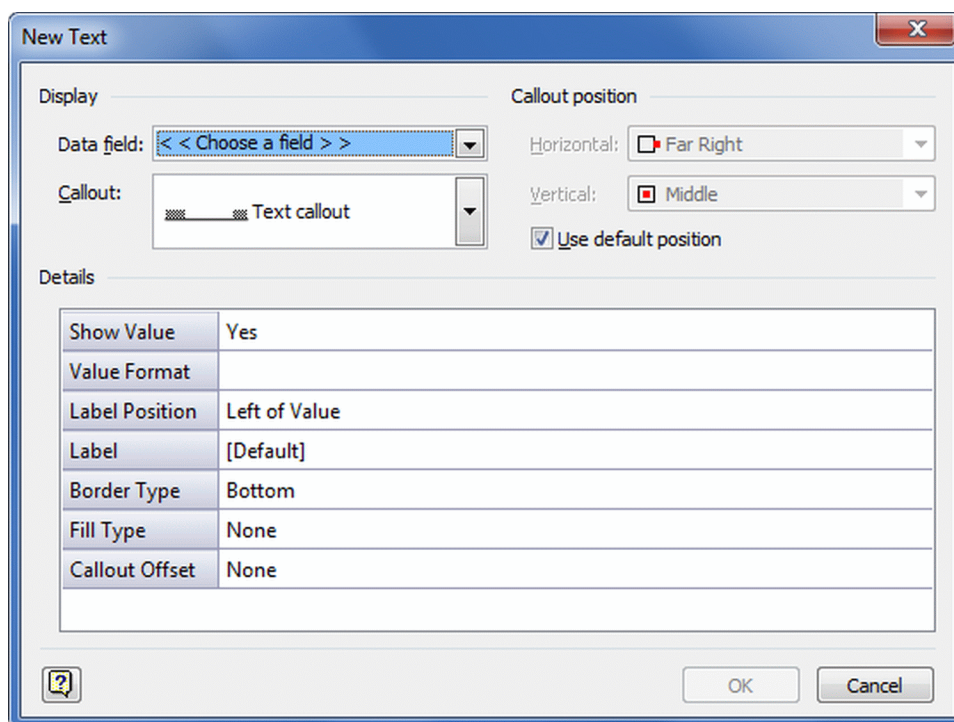
Currently the **Data Field** list is empty because no Shape Data has been associated to any data graphic element yet. The **Default position** setting on the lower left can be changed and applies to all listed **Data Field** items unless you disable it in the individual page setting and choose your own settings manually.

¹⁸⁵ I will show you how to make use of the color-by-value data graphic in a later section.

Clicking on the [New Item](#) button will activate a drop down menu showing the four data graphics type to choose from. Since we're interested in text callout, let's choose the [Text...](#) option.



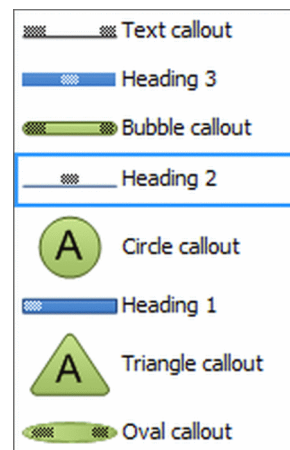
The New Text dialog box appears, prompting you by default to choose a field for the [Data field](#) item. If you click on the down arrow, you will see a list of the Shape Data's labels that you've entered for the resistor layout symbol, namely REFDES, PARTNO and VALUE.



Select REFDES since we want to display the reference designator for the resistor layout symbol.

The [Callout](#) item offers a variety of text callout format that you can choose to display your Shape Data's value. We want the resistor's REFDES value to be right in the center of the layout symbol without any fanfare and Shape Data label. You'll need to follow the steps below to achieve this:

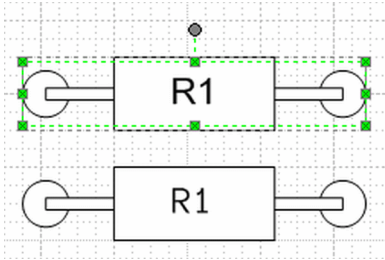
1. Select [Heading 2](#). The [Details](#) item will change accordingly, displaying Value Format, Border Type, Fill Type, Horizontal and Vertical Offsets properties. Set Border Type to [None](#) and leave the rest as default.
2. Uncheck the [Use default position](#) for the [Callout position](#) item and set the [Horizontal](#) item to [Center](#). Click OK.
3. Click OK again to complete and exit the Edit Data Graphic dialog box.



Our resistor layout symbol now shows the value of the REFDES Shape Data (right), which is right in the center of the symbol with the default Arial 9 point black color font belonging to the [Heading 2](#) text callout.

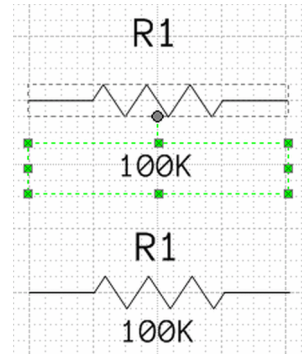


Question: Can we change the font? If you search the [New Text](#) dialog box which you use to configure the properties of [Heading 2](#) earlier on, seemingly there's no option to allow you to change the display font. If you recall, a data graphic is also a smartshape in itself and therefore should be editable like any other shape in Visio.



So to change the font, click on the resistor layout symbol, pause for a while then click on the R1 text. Notice the handles of the selected text callout have crosses inside compared to the plain handles on the first click? (Top) This indicates that the text callout is a sub-shape, just like those found in a grouped entity. You can now change the font using the formatting toolbar to another typeface, such as my favorite— Anonymous, 6 point (bottom).

Similarly, you can apply data graphics to the resistor's schematic symbol using the same process above, but this time you'll want to include the VALUE as well, so that REFDES's position is [Center, Above Shape](#) and VALUE is [Center, Below Shape](#).



Notice that the REFDES and VALUE are spaced a little too far apart in the original. No problem, we can sub-select, re-size and move them individually to the position we want just as easily like we did with the layout symbol using the sub-shape select.

When we're satisfied with the shape, we can duplicate it, right click to bring up the context menu and select [Data](#) → [Shape Data...](#) which will then allow us to enter a different REFDES and VALUE for the new resistor:

This is also a more intuitive and advanced method of adding new components to a schematic or layout diagram compared to the grouped entity method espoused in [Chapters 3](#) and [4](#), though both are just as acceptable, the difference being the former contains shape data that can be manipulated for a variety of useful purpose, while the latter is just a static shape. Of course, you'll have to be prepared to spend more time upfront to create and manage these smartshapes in order to tap into their power—that's the trade-off you need to consider. After all, there's no such thing as a free lunch!¹⁸⁶

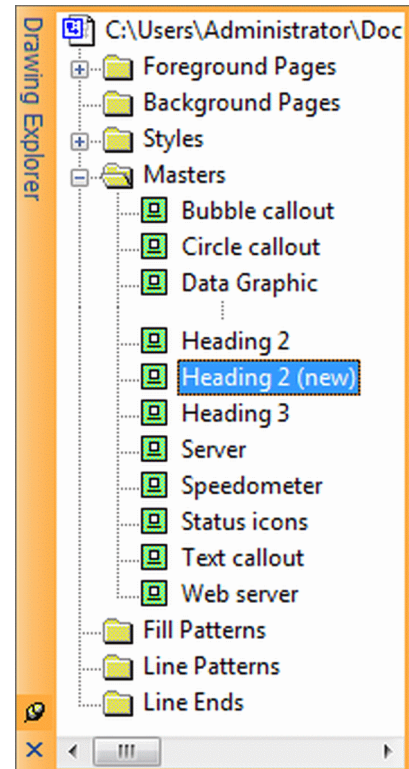
Customizing Your Own Data Graphics

Since data graphics are smartshapes, you can certainly customize or create your own. Customizing, though, would be the easier option. Let's look at how we can do it for the [Heading 2](#) text callout we used earlier.

To change the property of any shape or its associated data graphics, you need to first bring out drawing explorer from [View](#) → [Drawing Explorer Window](#). This will show you all the shapes and formatting associated with the Visio drawing file you're currently working on.

Now click on Masters to expand it and you'll find [Heading 2](#) among the list of shape stencils. You can double-click it to edit its properties in its own Visio drawing page but that will affect all shapes that use this text callout stencil. A better way is to duplicate a copy of it and rename the copy ([new](#)), and work on that copy instead.

After you've changed the font formatting for the new data graphic stencil which you've edited from the copy of the original [Heading 2](#), you can then open the [New Text](#) dialog box for the resistor layout symbol, and select [Heading 2 \(new\)](#) in the [Callout](#) drop down menu to effect the change.



Note:

This sub-section is really an after-thought to bring to my readers' awareness of the [Drawing Explorer Window](#) and its use, as well as the ability to customize data graphics to effect changes across the whole drawing, if required. I have intentionally left out the step-by-step details as I'm quite certain you should have no problem figuring out how to go about doing it—if you know Visio well enough by now.

¹⁸⁶ Hopefully, what you've seen so far has impressed you sufficiently to consider taking the better of the two approaches, even if it's not an immediate decision. I'm confident that after you're done with this chapter, you'll never be content to settle for anything less!

ShapeSheets

If you dislike Excel or struggle with spreadsheets, you may be tempted to skip this portion altogether. But I must warn you that if you do back out at this point, your knowledge of Visio will be incomplete. Let me make you a proposition here: stick through the fun and I assure you that you'll not only be amply rewarded—who knows, you might even change your mind about Excel too!

Alright, what is a ShapeSheet anyway? A picture speaks a thousand words so we'll begin with a simple example. Do the following steps:

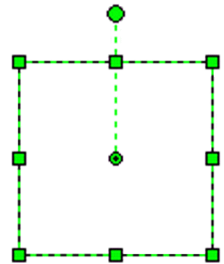
1. Open a blank Visio drawing page and draw a white-filled 1" x 1" square with the lower left corner at X=1" and Y=1". Leave the shape selected.
2. Next, go to [Window](#) → [Show ShapeSheet](#). The ShapeSheet property window opens up below the drawing page, dividing the screen in two.

If you were to scroll down and count the number of categories—or more correctly—[sections](#), you'll find that there are 14 of them just for this simple square which you've created, as shown overleaf. Feeling overwhelmed already? Don't worry, because we do not need to have to deal with each and every one of them directly most of the time, only the ones that require us to work at this level (and thankfully not very often—unless you are in the business of developing customized solutions¹⁸⁷).

The 14 sections are listed as follows:

- | | |
|--------------------|-----------------------|
| 1. Shape Transform | 8. Paragraph |
| 2. Geometry 1 | 9. Tabs |
| 3. Protection | 10. Text Block Format |
| 4. Miscellaneous | 11. Events |
| 5. Line Format | 12. Image Properties |
| 6. Fill Format | 13. Glue Info |
| 7. Character | 14. Shape Layout |

These are the sections that are visible and applicable to the square shape we just drawn. Interpreting the information is quite straightforward if you've followed my explanation on shapes closely. For example, the [Shape Transform](#) section shows the dimension of the square, the Pin Pos current position and its location on the shape, which is Width*0.5 and Height*0.5, or in other words [center-center](#). FlipX and FlipY are FALSE meaning no shape transformation is applied to the square.



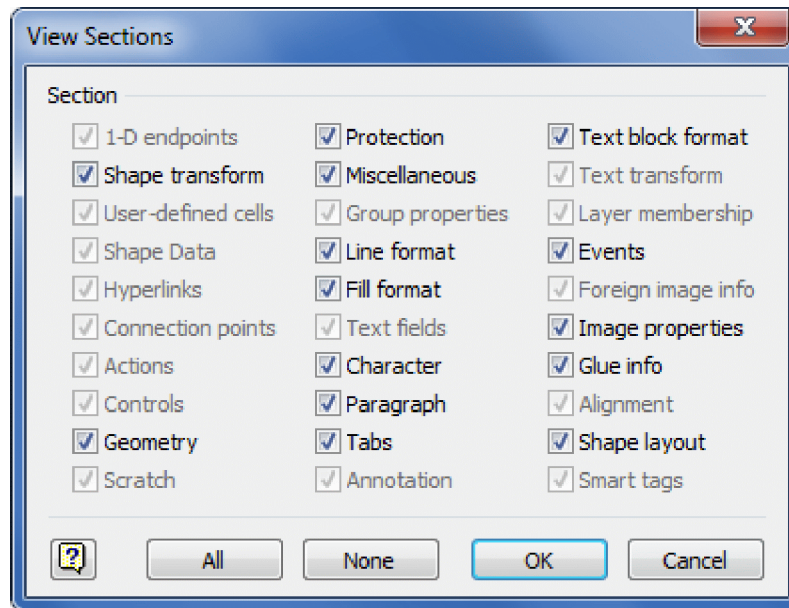
¹⁸⁷ If you happen to be in this unenviable position, I strongly recommend that you download a free copy of Visio 2007 software development kit (SDK) from Microsoft's website. It comes with a full off-line reference to every cell, formula, function and more within the ShapeSheet environment. The complete Visio Automation Object Model Reference is included as a bonus as well.

Mastering Visio's smartshape development and creating custom data graphics based on custom smartshapes is not a small undertaking for the faint-hearted, so if you're seriously thinking of going into it, then get a good book on ShapeSheet programming and if you can afford it, enroll for a certified training course too.

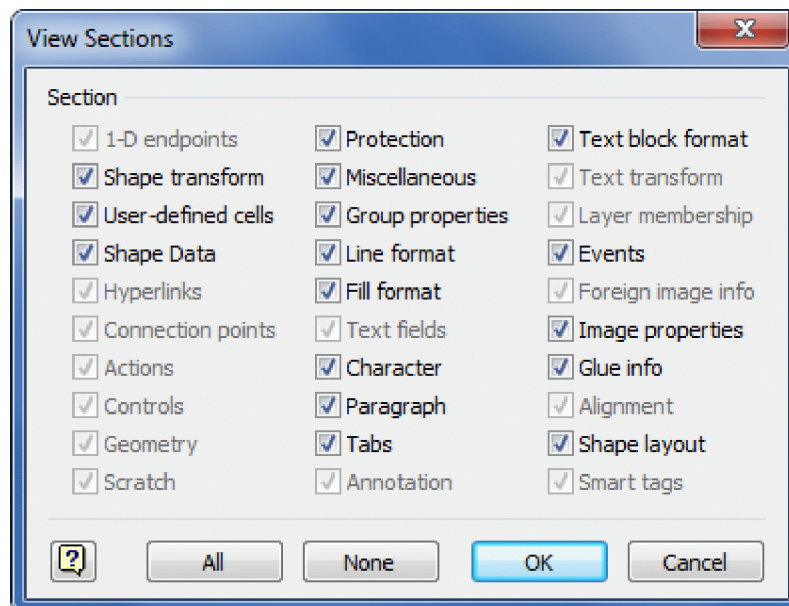
Shape Transform										
Width	1 in	PinX	1.5 in	FlipX	FALSE					
Height	1 in	PinY	1.5 in	FlipY	FALSE					
Angle	0 deg	LocPinX	Width*0.5	ResizeMode	0					
		LocPinY	Height*0.5							
Geometry 1										
Geometry1.NoFill		FALSE	Geometry1.NoLine		FALSE	Geometry1.NoShow		FALSE	Geometry1.NoSnap	FALSE
Name	X	Y	A	B	C	D	E			
1	MoveTo	Width*0	Height*0							
2	LineTo	Width*1	Height*0							
3	LineTo	Width*1	Height*1							
4	LineTo	Width*0	Height*1							
5	LineTo	Geometry1.X1	Geometry1.Y1							
Protection										
LockWidth	0	LockEnd	0	LockCrop	0					
LockHeight	0	LockDelete	0	LockGroup	0					
LockAspect	0	LockSelect	0	LockCalcWH	0					
LockMoveX	0	LockFormat	0	LockFromGroupFormat	0					
LockMoveY	0	LockCustProp	0	LockThemeColors	0					
LockRotate	0	LockTextEdit	0	LockThemeEffects	0					
LockBegin	0	LockVbEdit	0							
Miscellaneous										
NoObjHandles	FALSE	HideText	FALSE	ObjType	0					
NoCtlHandles	FALSE	UpdateAlignBox	FALSE	IsDropSource	FALSE					
NoAlignBox	FALSE	DynFeedback	0	Comment	**					
NonPrinting	FALSE	NoLiveDynamics	FALSE	DropOnPageScale	100%					
LangID	1033	Calendar	0	LocalizeMerge	FALSE					
Line Format										
LinePattern	1	BeginArrow	0	BeginArrowSize	2					
LineWeight	0.24 pt	EndArrow	0	EndArrowSize	2					
LineColor	0	LineColorTrans	0%	Rounding	0 in					
LineCap	0									
Fill Format										
FillForegnd	1	ShdwForegnd	0	ShapeShdwOffsetX	0 in					
FillForegndTrans	0%	ShdwForegndTrans	0%	ShapeShdwOffsetY	0 in					
FillBkgnd	0	ShdwBkgnd	1	ShapeShdwType	0					
FillBkgndTrans	0%	ShdwBkgndTrans	0%	ShapeShdwObliqueAngle	0 deg					
FillPattern	1	ShdwPattern	0	ShapeShdwScaleFactor	100%					
Character	Font	Size	Scale	Spacing	Color	Transparency	Style	ComplexScriptSize	LangID	
0	4	12 pt	100%	0 pt	0	0%	0	-100%	1033	
Paragraph	IndFirst	IndLeft	IndRight	SpLine	SpBefore	SpAfter	HAlign	BulletSize	Flags	
0	0 in	0 in	0 in	-120%	0 pt	0 pt	1	-100%	0	
Tabs	Position	Alignment	Position	Alignment	Position	Alignment				
0										
Text Block Format										
LeftMargin	4 pt	TopMargin	4 pt	TextDirection	0					
RightMargin	4 pt	BottomMargin	4 pt	VerticalAlign	1					
TextBkgnd	0	TextBkgndTrans	0%	DefaultTabStop	0.5 in					
Events										
TheData	No Formula	EventDbClick	No Formula	EventDrop	No Formula					
TheText	No Formula	EventXfMod	No Formula	EventMultiDrop	No Formula					
Image Properties										
Contrast	50%	Gamma	1	Sharpen	0%					
Brightness	50%	Blur	0%	Denoise	0%					
Transparency	0%									
Glue Info										
BegTrigger	No Formula	GlueType	0	WalkPreference	0					
EndTrigger	No Formula									
Shape Layout										
ShapePermeableX	FALSE	ShapePermeableY	FALSE	ShapePermeablePlace	FALSE					
ShapeFixedCode	0	ShapePlowCode	0	ShapeRouteStyle	0					
ConLineJumpDirX	0	ConLineJumpDirY	0	ConFixedCode	0					
ConLineJumpCode	0	ConLineJumpStyle	0	ShapeSplit	0					
ShapePlaceFlip	0	ConLineRouteExt	0	ShapeSplittable	0					
ShapePlaceStyle	0									

Figure 5.11 Sections of a ShapeSheet

Within the ShapeSheet windows, you can right click on any gray area and select [View Sections...](#) from the context menu, or go to [View](#) → [Sections...](#) on the main window menu to bring up the [View Sections](#) dialog box:



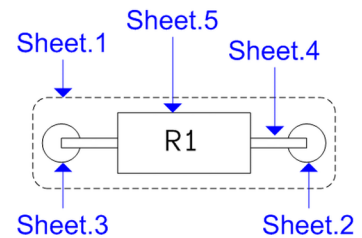
You'll notice that there are 14 active sections you can selectively view or hide, while the remaining 16 sections are grayed out because they are not relevant to our square shape at this point. For a more complicated shape like our resistor layout symbol, which contains four grouped sub-shapes as well as shape data and applied data graphics, the number of sections will be more and vary according to the shape properties:



Unlike the [Shape Data](#) and [Size & Position windows](#) which update the display information of each shape you click on, the [ShapeSheet](#) window will display the information of the selected shape only when you do a right click on it and choose [Show ShapeSheet](#) from the context menu or go to [Windows](#) → [Show ShapeSheet](#) from the main menu. You can also view the ShapeSheet of a sub-shape within a grouped object using this method.

Coming back now to our resistor layout symbol, in order to manipulate the individual sub-shapes within the grouped shape (parent), we'll need to first find out their IDs to reference them. One way is to click on the parent shape and then each sub-shapes in turn (body, lead, left and right pads), and perform a [Windows](#) → [Show ShapeSheet](#) each time, taking reference from the ShapeSheet's window title which will display something like <Drawing Title>:Page-?:Sheet.? where [Page-?](#) is the active drawing page your shape is on, and [Sheet.?](#) is the ID of the parent or sub-shape that is selected.

Depending on how you draw your resistor layout symbol, which page you're on, and whether there's any other shapes on that page, the IDs will differ. In our case, let's assume that the resistor layout symbol is the only shape on the page with IDs as shown on the right. We are interested in the Shape Transform and Shape Data sections for the group (parent) shape, and just the Shape Transform section for the sub-shapes:



Shape Transform					
Width	1.1719 in	PinX	4.4766 in	FlipX	FALSE
Height	0.2494 in	PinY	6.5003 in	FlipY	FALSE
Angle	0 deg	LocPinX	Width*0.5	ResizeMode	0
		LocPinY	Height*0.5		

Group (Parent)

Shape Data	Label	Prompt	Type	Format	Value	SortKey	Invisible	Ask	LangID	Calendar
Prop.Ref_Des	"REFDES"	No Formula	0	No Formula	"R1"	No Formula	No Formula	No Formula	1033	No Formula
Prop.Part_No	"PARTNO"	No Formula	0	No Formula	"RNC55H1003FSCJ"	No Formula	No Formula	No Formula	1033	No Formula
Prop.Res_Val	"VALUE"	No Formula	0	No Formula	"100K"	No Formula	No Formula	No Formula	1033	No Formula

Shape Transform					
Width	Sheet.1!Width*0.1351	PinX	Sheet.1!Width*0.9324	FlipX	FALSE
Height	Sheet.1!Height*0.635	PinY	Sheet.1!Height*0.5	FlipY	FALSE
Angle	0 deg	LocPinX	Width*0.5	ResizeMode	0
		LocPinY	Height*0.5		

(Right Pad)

Shape Transform					
Width	Sheet.1!Width*0.1351	PinX	Sheet.1!Width*0.0676	FlipX	FALSE
Height	Sheet.1!Height*0.635	PinY	Sheet.1!Height*0.5	FlipY	FALSE
Angle	0 deg	LocPinX	Width*0.5	ResizeMode	0
		LocPinY	Height*0.5		

(Left Pad)

Shape Transform					
Width	Sheet.1!Width*0.8649	PinX	Sheet.1!Width*0.5	FlipX	FALSE
Height	Sheet.1!Height*0.16	PinY	Sheet.1!Height*0.5	FlipY	FALSE
Angle	0 deg	LocPinX	Width*0.5	ResizeMode	0
		LocPinY	Height*0.5		

(Lead)

Shape Transform					
Width	Sheet.1!Width*0.4682	PinX	Sheet.1!Width*0.5	FlipX	FALSE
Height	Sheet.1!Height*1	PinY	Sheet.1!Height*0.5	FlipY	FALSE
Angle	0 deg	LocPinX	Width*0.5	ResizeMode	0
		LocPinY	Height*0.5		

(Body)

Only the group (parent) shape contains shape data because they're added to it after the sub-shapes were grouped. Now to effect the change in geometry in each of the sub-shapes based on the wattage of the resistor layout symbol, we need to add a new shape data called RATING to the group shape:

1. Select the resistor layout symbol, right click and choose **Data** → **Shape Data...** to bring up the Shape Data dialog box, then click the **Define...** button to bring up the Define Shape Data dialog box:

The 'Define Shape Data' dialog box is shown with the following fields and settings:

- Label:** RATING
- Name:** Rating
- Type:** Fixed List (dropdown)
- Language:** English (United States) (dropdown)
- Format:** 0.125; 0.25; 0.5; 1 (with a right arrow button)
- Calendar:** (empty dropdown)
- Value:** (empty text field)
- Prompt:** Select a rating.
- Sort key:** (empty text field)
- ☐ Ask on drop
- ☐ Hidden

Properties:

Label	Name	Type
PARTNO	Part_No	String
VALUE	Res_Val	String
RATING	Rating	Fixed List

At the bottom are buttons: New, Delete, OK, and Cancel.

2. Enter the data as shown above. Take note that the **Type** is a **Fixed List** and the **Format** entry is given as a list of four numbers separated by the ';' delimiter. Leave the **Value** field empty for now as it will be automatically filled a while later. Enter 'Select a rating' in the **Prompt** field. Click **OK** to return back to the **Shape Data** dialog box.

You'll now see an additional field which is a drop down menu containing the four ratings for selection and the **Prompt** (slightly obscured by the drop down menu) telling you to select a rating value when you place your cursor in the RATING field or click on the down arrow to view the available choices. Neat huh?

The Shape Data dialog box contains the following fields and options:

- REFDES: R1
- PARTNO: RNC55H1003FSCJ
- VALUE: 100K
- RATING: 0.25 (with a dropdown arrow)
- Prompt: Select a rating (with a list of options: 0.125, 0.25, 0.5, 1)

Buttons at the bottom: Define..., OK, Cancel.

You can select any of the rating values but the shape will not change its geometry to reflect the rating—yet. We still need to link the RATING shape data to the Shape Transform section of each of the sub-shapes to effect the change.

When you select the group (parent) shape and bring up its Shapesheet, you'll see the additional RATING shape data item:

Shape Data	Label	Prompt	Type	Format	Value
Prop.Ref_Des	"REFDES"	No Formula	0	No Formula	"R1"
Prop.Part_No	"PARTNO"	No Formula	0	No Formula	"RNC55H1003FSCJ"
Prop.Res_Val	"VALUE"	No Formula	0	No Formula	"100K"
Prop.Rating	"RATING"	"Select a rating."	1	"0.125; 0.25; 0.5; 1"	INDEX(3,Prop.Rating.Format)

This is the [Formulas](#) view of the Shapesheet; you can change it to the [Values](#) view simply by right clicking anywhere on the gray area of the Shapesheet and select [Values](#). It will then display the actual value instead of formula. We want to stick with the [Formulas](#) view for now.

Under the [Shape Data](#) heading you'll see the names which we've entered with a 'Prop' prefixed to denote property names. The value for [Prop.Rating](#) is shown as INDEX(3,Prop.Rating.Format) which implies that it is currently at the value '1'. Indexing is numbered from 0 to 3 for the four ratings under the [Format](#) field. If you right click on the group (parent) shape and choose [Data](#) → [Shape Data...](#) to bring up the [Shape Data](#) dialog box, then select the [0.25](#) rating, the [Prop.Rating](#) value will change to INDEX(1,Prop.Rating.Format) when you click [OK](#).

Let's link [Prop.Rating](#) to the Shape Transform section of each sub-shape now...

3. Select the body sub-shape and open its Shapесheet.

Shape Transform					
Width	Sheet.1!Width*0.4682	PinX	Sheet.1!Width*0.5	FlipX	FALSE
Height	Sheet.1!Height*1	PinY	Sheet.1!Height*0.5	FlipY	FALSE
Angle	0 deg	LocPinX	Width*0.5	ResizeMode	0
		LocPinY	Height*0.5		

If you compare the Shape Transform section of the group (parent) shape to its sub-shapes, you'll notice that the group (parent) shape shows fixed values for its **Width**, **Height**, **PinX** and **PinY** fields, while the sub-shapes only show formulas like the ones you see above. This means that all sub-shapes take reference from their parent shape, such that if you resize or move the group (parent) shape, Visio will calculate the values for each of the sub-shapes based on the new dimensions or position of the group (parent) shape.

To reference the **Prop.Rating** parameter from any sub-shape, you'll need to precede the name with the ID of the parent shape, and in this case it is 'Sheet.1' followed by an '!' (as in the above section) so the full reference is Sheet.1!Prop.Rating.

So how do we let the sub-shape know what **Prop.Rating** value is selected? Visio's Shapесheet comes with a rich set of functions that let you to perform logical and mathematical operations, as well as make decisions based on conditions which you set using the IF statement. And this is just what we need. The syntax is:

IF (logicalexpression, valueiftrue, valueiffalse)

where if the logical expression is true, it will return valueiftrue, otherwise valueiffalse. You can nest multiple IF statements within a single line, so converting the body length and diameter values on [page 205](#) from mm to the nearest decimal inches, we can replace the **Width** and **Height** formulas for the body sub-shape by selecting their fields or cells in turn and entering the following statements:

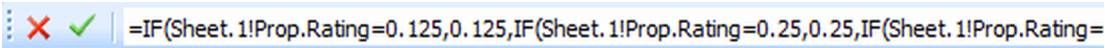
Width:

IF(Sheet.1!Prop.Rating=0.125,0.125,IF(Sheet.1!Prop.Rating=0.25,0.25,IF(Sheet.1!Prop.Rating=0.5,0.35,0.43)))

Height:

IF(Sheet.1!Prop.Rating=0.125,0.07,IF(Sheet.1!Prop.Rating=0.25,0.098,IF(Sheet.1!Prop.Rating=0.5,0.125,0.2)))

into the Shapесheet formula bar and click on the green tick to accept or red cross to cancel the change:



Now when you bring up the Shape Data dialog box and select a value for the rating, you'll see the geometry of the resistor's body transform (i.e. changes its size) accordingly.

4. We can repeat the same process for the resistor's lead by selecting the lead sub-shape and bringing up its Shapesheet, then enter the following statements into the Shapesheet formula bar for the **Width** and **Height**:

Width:

```
IF(Sheet.1!Prop.Rating=0.125,0.35,IF(Sheet.1!Prop.Rating=0.25,0.5,IF(Sheet.1!Prop.Rating=0.5,0.6,0.8)))
```

Height:

```
IF(Sheet.1!Prop.Rating=0.125,0.02,IF(Sheet.1!Prop.Rating=0.25,0.025,IF(Sheet.1!Prop.Rating=0.5,0.025,0.032)))
```

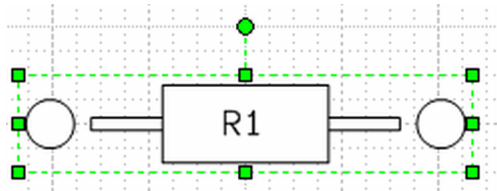
5. How about the resistor pads? Let's give it a try and see how it goes. First, select the left pad sub-shape and bring up its Shapesheet, then enter the following statement into the Shapesheet formula bar for the **Width** and **Height**:

Width and Height:

```
IF(Sheet.1!Prop.Rating=0.125,0.075,IF(Sheet.1!Prop.Rating=0.25,0.085,IF(Sheet.1!Prop.Rating=0.5,0.1,0.125)))
```

Next, select the right pad sub-shape and bring up its Shapesheet, then enter the same statement above into the Shapesheet formula bar for the **Width** and **Height**. They should be similar pad size pairs, remember?

Now take a look at the group (parent) shape:



Oops... what have we here? Something's certainly not right. Instead of conforming to the lead length for their spacing, both pads remain where they are even though their sizes change to the tune of the Prop.Rating parameter. This indicates that there is a problem with the X-axis transpositions of the left and right pads. If we look at the PinX and PinY of the Shape Transform sections of both pads, we notice that they reference the geometry of the group (parent) shape as follows:

Left pad	Sheet.1!Width*0.0676	Sheet.1!Height*0.5
Right pad	Sheet.1!Width*0.9324	Sheet.1!Height*0.5

Cross-referencing the group (parent) shape, it became evident that the **Width** and **Height** values are constants, so based on the formulas above, **PinX** will logically remain fixed in a constant position as well. We didn't encounter this problem with the body and lead because they both took reference to the centre of the group (parent) shape, but it did not work out for

the left and right pads which are offset from the center of the group (parent) shape in the X-axis. Now do you see the problem?

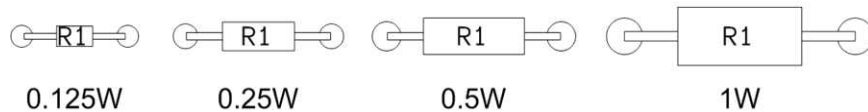
Fortunately the solution is a simple one, and that is to make the [Width](#) of the group (parent) shape variable based on the ratings as well. Referring to the pad diameter and spacing data on [page 205](#), we can select the group (parent) shape and bring up its Shapesheet, then enter the following statements into the Shapesheet formula bar for the [Width](#) and [Height](#):

Width:

`IF(Prop.Rating=0.125,0.425,IF(Prop.Rating=0.25,0.585,IF(Prop.Rating=0.5,0.7,0.925)))`

Note that there's no need for the Sheet.1! prefix since the Prop.Rating parameter belongs to the group (parent) shape anyway.

Now when we select any one of the four ratings, we'll get the correct geometrical adjustments for all the sub-shapes in the resistor layout symbol:



The 0.125W resistor layout symbol looks pretty cramped for the reference designator to fit in comfortably and will certainly fall outside the boundary if it's three or more characters (e.g. R10, R100, etc.). Is there a solution to this problem? Yep, there sure is but I'm not going to serve it up on a silver platter (grin!). Instead, I want to let you have a go at solving it yourself, but I'll drop you some hints here:

- Remember the reference designator is a data graphic object, which is also a smartshape in itself.
- Since it's congested inside the body, you may want to bring the reference designator outside. You only need to use just one IF statement for the 0.125 rating value to effect the change in the [PinY](#) formula for the data graphic object.

And that pretty much sums up the discussion on the three shape musketeers—Shape Data, Shape (Data) Graphics, and Shapesheets. Now that we've laid some foundations, we are ready to take a look at two peculiar yet interesting type of smartshapes...

Building a Multi-Field Shape

Utilizing the knowledge you've just gained about smartshapes, it is not difficult to construct a multi-field shape such as a title block which, once completed and saved as a master stencil, will automatically fill up the necessary fields based on the document you're working on and prompt you for data that are outside of its scope, when you drag and drop a copy of it into your drawing's background page.

Ø	2015-01-06	ORIGINAL	RANDY DM
REV	DATE	AMENDMENTS	APPROVED
DRAWING TITLE: TRACKING RADAR INTERFACE BOARD			
CUSTOMER: INTELLi-SYSTEMS		 Kinetec Corp. A company of Associated Engineers in Electronics	
DRAWN BY: NG KENG TIONG		JOB NO: 2014-12-0386	SUB-J/NO: 0386-001
REVIEW BY: RANDY DYKEMAN		DRAWING NO: LD123456-050	REV: Ø SHEET: 1 OF 2

DRAWING TITLE: TRACKING RADAR INTERFACE BOARD			
DRAWN BY: NG KENG TIONG		JOB NO: 2014-12-0386	SUB-J/NO: 0386-001
REVIEW BY: RANDY DYKEMAN		DRAWING NO: LD123456-050	REV: Ø SHEET: 2 OF 2

Figure 5.12 Main and sub-title blocks with multiple field entries.

Consider the title block templates shown above: (Top) a main title block with greater details (company logo, customer, amendments history) on the first page, and (Bottom) a sub-title block with just the essentials on the rest of the pages.¹⁸⁸ I'll focus on creating the main title block and leave the simpler sub-title block as an additional exercise for my readers. Again, planning is important if you want to get things done the proper way. Looking at the main title block, we can determine the fields which will be automatically filled:

- Drawing title Title of the document
- Drawn by Author (Creator)
- Sheet number Page and Number of pages
- Date Creation date

¹⁸⁸ This is a more sophisticated and professional example of a title block compared to the one we've created in Chapter 3 for the SCSCI host adapter layout diagram which does not make use of any smartshape features at all.

Don't worry if you're feeling lost or wondering where to obtain those data. Things will clear up when we get to the relevant portions. In fact, after going through the exercise, you'll have a better understanding about Microsoft Office's components, why they're there and how you can make use of the information they contain in meaningful ways.

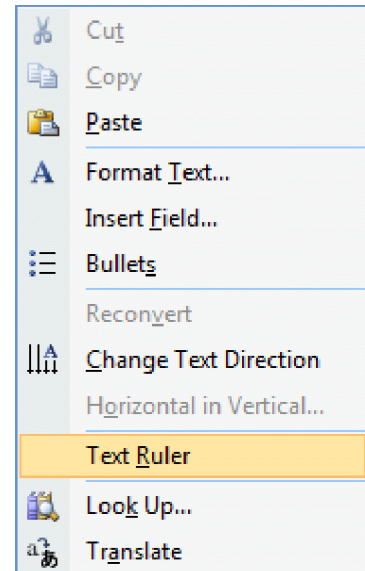
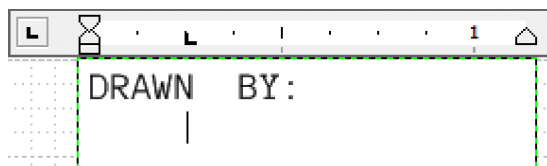
Before you can build a multi-field shape, you'll need to be aware that Visio provides a means to insert pre-defined or custom data as text into selected shapes. Let's do a simple example so you know what I mean by that statement:

1. Create a **1.1875 x 0.2813** white-filled rectangle on an empty page and format its text properties as follows:

Font	Anonymous, Regular, 6 pt.
Paragraph	Left align
Text block	Alignment : Top
	Margins : Top, Bottom, Left, Right = 2 pt.

2. With the shape still selected, type in 'DRAWN BY:' (without the quotes, two spaces in between words) follow by an ENTER to go to the next line¹⁸⁹ and then press the TAB key once.
3. Next, we want to bring up the text ruler to adjust the position of the tab spacing in the text block. While still in text edit mode, right click to bring up the context menu (right) and select **Text Ruler**. The text ruler will now appear whenever you're in the text edit mode.

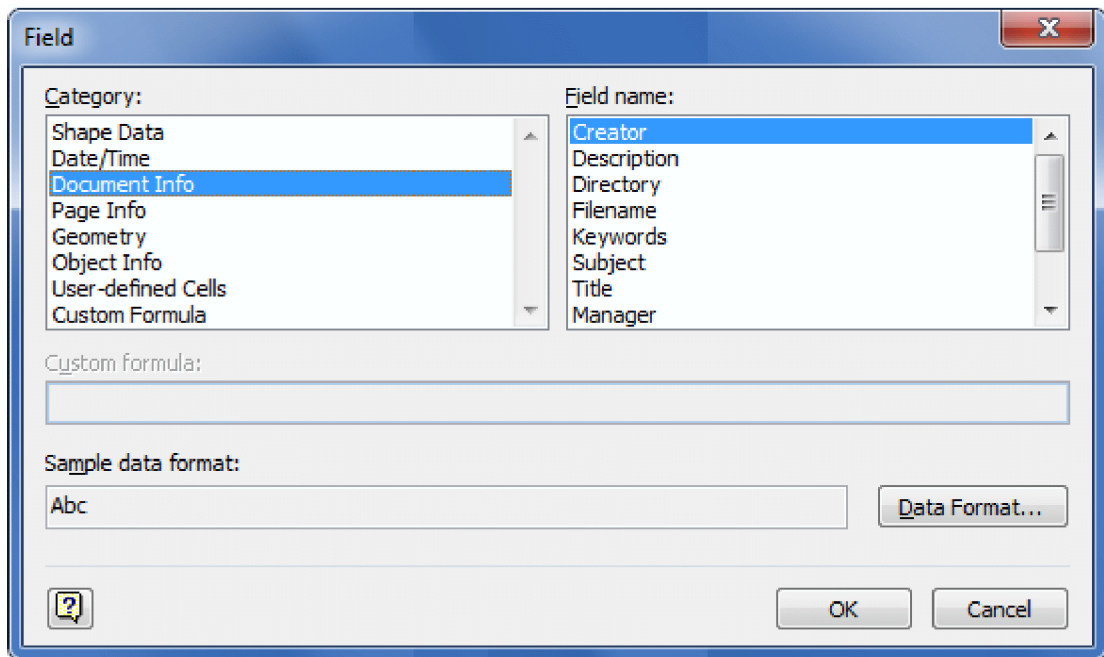
Notice that there's an 'L' symbol to the left of the text ruler, which is the tab alignment toggle button that's currently set to left align. Click on the ruler at the 0.25" mark and you'll see the tab spacing jump to the new indentation:



This is a very useful way to align your text within the text block which makes the presentation much neater, especially when you have multiple field entries in one shape. We now have the field name (**Drawn By:**) keyed in manually and we want to add in the field value, which in this instance is the author or creator of the drawing. This brings us to the next important component of Visio...

¹⁸⁹ When you start typing into a selected shape, you effectively enter into the text edit mode. Pressing the ENTER key inside the text block brings you to the next line. Visio supports multiple lines text entry.

4. In text edit mode with the cursor at the tab spacing as shown above, go to [Insert](#) → [Field...](#) The Field dialog box appears. Select the Document Info and click on the Creator field name as shown below:



Click OK and the field value with my name 'NG KENG TIONG' will be appended at the tab spacing where the cursor is:

DRAWN BY:
NG KENG TIONG

To understand where Visio gets the Creator information from, first take a closer look at the field names available in the Document Info category (you'll need to scroll down to see the complete listing). Got the clue? That's right, if you go to [File](#) → [Properties...](#) you'll see almost all the items in the field name list (see figure overleaf), except for Directory and Filename. The Creator field name is synonymous to Author (don't ask me why Visio chose to use a different name) and the same capitalization is reflected in the above text block when you insert it at the tab spacing. If you change the Author field in the file properties dialog box and click OK, it will be immediately updated in the text block containing the Creator entry, including changes to the letter casing! Are you beginning to see the advantage of using multi-field shapes to add that extra dimension of smartness to your Visio drawing now?

There's more in store, but before we move on, let's list out all the category and field names in the Field dialog box so you get a full picture of the information that's available at your disposal, and some (refer to next page).

General

Summary

Contents

Title:

TRACKING RADAR INTERFACE BOARD

Subject:

PCB Layout Diagram

Author:

NG KENG TIONG

Manager:

Randy Dykeman

Company:

ST Electronics Ltd

Language:

English (United States)

Category:

Keywords:

Description:

Hyperlink base:

☒ Save preview picture

☒ Save workspace

Table 5.1 Field dialog box's Category and their Field name lists.

Category:	Field name:
Shape Data	
Date/Time	Creation, Current, Last Edit and Print Date/Time
Document Info	Creator, Description, Directory, Filename, Keywords, Subject, Title, Manager, Company, Category, Hyperlink Base
Page Info	Background, Name, Number of Pages, Page Number
Geometry	Width, Height, Angle
Object Info	Data 1, Data 2, Data 3, ID, Master, Name, Type
User-defined Cells	
Custom Formula	=

The Shape Data, User-defined Cells and Custom Formula categories are empty because none has been defined for these yet.

Referring to the main title block in Figure 5.12, let's create the rest of the text blocks to construct the following template shell using the same methodology as the 'DRAWN BY:' text block which we've done previously:

Figure 5.13 Completed template shell for the main title block.

Table 5.2 Main title block component specifications.

Item	Name	Width	Height	Tab Spacing	
				Mark	Align
1	REVIEW BY:	1.1875	0.2813	0.25	Left
2	DRAWN BY:	1.1875	0.2813	0.25	Left
3	CUSTOMER:	1.1875	0.4063	0.5625	Center
4	DRAWING NO:	1.4375	0.2813	0.625	Left
5	JOB NO:	1.4375	0.2813	0.625	Left
6	REV:	0.3750	0.2813	0.25	Center
7	SHEET:	0.8125	0.2813	0.375	Left
8	SUB-J/NO:	1.1875	0.2813	0.625	Left
9	Company Logo	2.6250	0.4063	-	-
10	DRAWING TITLE:	3.8125	0.2813	2.00	Center
11	REV	0.4375	0.1563	-	-
	DATE	0.7500	0.1563	-	-
	AMENDMENTS	2.0000	0.1563	-	-
	APPROVED	0.6250	0.1563	-	-

Unless otherwise specified, font used is Anonymous, regular, 6 pt.

A few things to take note:

- Single line or multi-line text blocks have single-line spacing paragraph settings.
- Company logo and amendments (document revisions) text blocks do not have tab spacings.
- Drawing Title's field value font will eventually be set to 8 pt. when inserted.
- The number of rows in the amendments block is up to your preference, and depends on how many revisions or engineering change order (ECO) may be expected or allowed before a new design ensues.
- You can include your company's logo in the reserved block at this time, either by copy and paste method using graphics, or if you prefer, create a Visio diagram of the logo with the advantage that it's vector-based and not bitmap, hence the logo will remain sharp even if you resize it.

Next, we're going to insert the fields that will automatically be filled up by Visio, namely the drawing title, reviewer, sheet number and date (the author or creator has already been done previously).

1. Double-click on the DRAWING TITLE: shape and place the cursor on the tab spacing in text edit mode. Go to [Insert](#) → [Field...](#) to bring up the Field dialog box, then select the Document Info category and its Title field name. Click OK. Select the whole field value (in this case, TRACKING RADAR INTERFACE BOARD) and change the font size to 8 pt.
2. Double-click on the REVIEW BY: shape and place the cursor on the tab spacing in text edit mode. Go to [Insert](#) → [Field...](#) to bring up the Field dialog box, then select the Document Info category and its Manager field name. Click OK. Notice that the reviewer's name is capitalized on the first letters only, so if we want the full name to be in uppercase, go to [File](#) → [Properties...](#) and type over the Manager's name in full capital letters, then click OK.¹⁹⁰
3. Double-click on the SHEET: shape and place the cursor on the tab spacing in text edit mode. Go to [Insert](#) → [Field...](#) to bring up the Field dialog box, then select the Page Info category and its Page number field name. Click OK. Next append ' OF ' to the back of the page number, then bring up the Field dialog box again and select the Number of pages field name in the Page Info category this time. Click OK.¹⁹¹
4. Select the blank shape above the DATE shape. Go to [Insert](#) → [Field...](#) to bring up the Field dialog box, then select the Date/Time category and its Last Edited Date/Time field name. Click OK.¹⁹²

¹⁹⁰ Do not change it at the shape's text block because it will only be a local change and will not be updated in the Manager's field in the Properties dialog box.

¹⁹¹ The SHEET: shape is a more complex multi-field shape in that it contains two sets of field names and values instead of one, which proves that you can have different combination of these elements to create sophisticated forms in Visio.

¹⁹² There is no need to double-click to enter into text edit mode since there's no tab spacing for this shape and we're using the default text formatting to place the Date/Time field value. You can choose the format in which the date (and time) to be displayed by clicking on the Data Format... button to bring up its dialog box to select the format (date only, date with time, etc.) to use. Note that Creation Date/Time is when the document file was first created and will remain constant, whereas Last Edited Date/Time is updated each time the document is saved after editing.

Now that we're done with the automatic part, we'll need to define shape data for the main title block so that when you drag and drop a copy from the master stencil, it will prompt you for these information and update the remaining fields in the title block. You can choose to define the rest of the fields as shape data, or select just the essential ones and fill up the remaining fields (AMENDMENTS, APPROVED, etc.) directly on the shape text blocks as and when necessarily.

Let's define the shape data for the following fields:

- Drawing number and Revision (Original)
- Customer
- Job number
- Sub-Job number

Before we proceed, we'll need to group the whole chunk of multi-field shapes and what have you into a single object parent entity. Once you're done, right click on the group (parent) shape and select Data → Shape Data... When prompted that no shape data exists and whether you want to define new shape data, click Yes. The Define Shape Data dialog box will appear and you can start defining the following shape data,¹⁹³ click OK, and then enter the respective values in the Shape Data dialog box:

Define Shape Data	Shape Data
DRAWING NO	LD123456-050
REVISION	0
CUSTOMER	INTELLi-SYSTEMS
JOB NO	2014-12-0386
SUB-JOB NO	0386-001

Once you're done, with the title block still selected, go to [Window](#) → [Show Shapsheet](#) to open up and view its Shapsheet:

Shape Data	Label	Prompt	Type	Format	Value	SortKey	Invisible	Ask
Prop.Row_1	"DRAWING NO"	No Formula	0	No Formula	"LD123456-050"	No Formula	No Formula	TRUE
Prop.Row_2	"REVISION"	No Formula	0	No Formula	"0"	No Formula	No Formula	TRUE
Prop.Row_3	"CUSTOMER"	No Formula	0	No Formula	"INTELLi-SYSTEMS"	No Formula	No Formula	TRUE
Prop.Row_4	"JOB NO"	No Formula	0	No Formula	"2014-12-0386"	No Formula	No Formula	TRUE
Prop.Row_5	"SUB-JOB NO"	No Formula	0	No Formula	"0386-001"	No Formula	No Formula	TRUE

In the Shape Data section of the title block's Shapsheet, you'll see the five shape data that you've defined together with their values. Notice that the Ask field for these shape data are TRUE indicating that the Shape Data dialog box will appear to prompt the user when a new title block is drag and drop into a drawing or background page from the master stencil (which you've yet to create from the multi-field title block you're currently working on now).

¹⁹³ You only need to key in the LABEL fields in the Define Shape Data dialog box and leave the rest empty. But remember to tick the 'Ask on drop' check box for each shape data you defined so that when a new title block is drag and drop from the master stencil, you will be prompted to enter these information.

If you select each of the individual sub-shapes that had field values already inserted before grouping and view their Shapeweets, you'll find some interesting data in their Text Fields section:

Text Fields	Format	Value	Calendar	ObjectKind
0	FIELDPICTURE(37)	TITLE()	0	0

Text Fields	Format	Value	Calendar	ObjectKind
0	FIELDPICTURE(37)	CREATOR()	0	0

Text Fields	Format	Value	Calendar	ObjectKind
0	FIELDPICTURE(37)	MANAGER()	0	0

Text Fields	Format	Value	Calendar	ObjectKind
0	FIELDPICTURE(0)	PAGENUMBER()	0	0
0	FIELDPICTURE(0)	PAGECOUNT()	0	0

Text Fields	Format	Value	Calendar	ObjectKind
0	FIELDPICTURE(200)	DOCLASTEDIT()	0	0

The Text Fields section displays functions and formulas which are inserted into the shape's text using the Field dialog box.¹⁹⁴ It contains the format function (FIELDPICTURE) which returns a format-picture string that matches Visio's internal text field format code (the number inside the braces), the purpose of which is to define the expansion of values to dates, times, numbers and unit labels.¹⁹⁵ What interests us is the Value field or cell which, if you have been following me this far, should start to make some sense to you now:

TITLE()	DRAWING TITLE:	Document Info	Title
CREATOR()	DRAWN BY:	Document Info	Creator
MANAGER()	REVIEW BY:	Document Info	Manager
PAGENUMBER()	SHEET: x	Page Info	Page Number
PAGECOUNT()	SHEET: OF y	Page Info	Number of Pages
DOCLASTEDIT()	DATE:	Date/Time	Last Edit

Hopefully that gets the light bulb switch on in you, more so for the benefit of your understanding and appreciation of text field properties and their connection with multi-field shapes. Let's finish up our title block with the remaining link ups for the shape data, shall we?

First, let's change the five shape data names in the group (parent) shape's Shape Data section from the Visio assigned generic forms (Prop.Row_x) to more specific, recognizable forms:

Prop.Row_1	⇒	Prop.DrawingNo	Note: Simply click on the 'Prop.Row_x' cells and type in the name without the 'Prop.' prefix; Visio will automatically append it for you.
Prop.Row_2	⇒	Prop.Revision	
Prop.Row_3	⇒	Prop.Customer	
Prop.Row_4	⇒	Prop.JobNo	
Prop.Row_5	⇒	Prop.SubJobNo	

¹⁹⁴ And that, incidentally, is the only way in which the Text Fields section is inserted into Shapeweets.

¹⁹⁵ Referenced from Visio's SDK documentation. Don't worry if you don't understand what that long sentence means; it's something internal to Visio and shouldn't concern the average user anyway.

Next, we will individually select each sub-shape, bring up its Shapesheet, insert a User-defined section and rename it to 'User.fieldValue', and type the shape data reference from the group (parent) shape into the Value cell prefixed by its shape ID which, if you remember, will be 'Sheet.1!'. The completed entry will look like the following:

User-defined Cells	Value	Prompt
User.fieldValue	Sheet.1!Prop.DrawingNo	""

User-defined Cells	Value	Prompt
User.fieldValue	Sheet.1!Prop.Revision	""

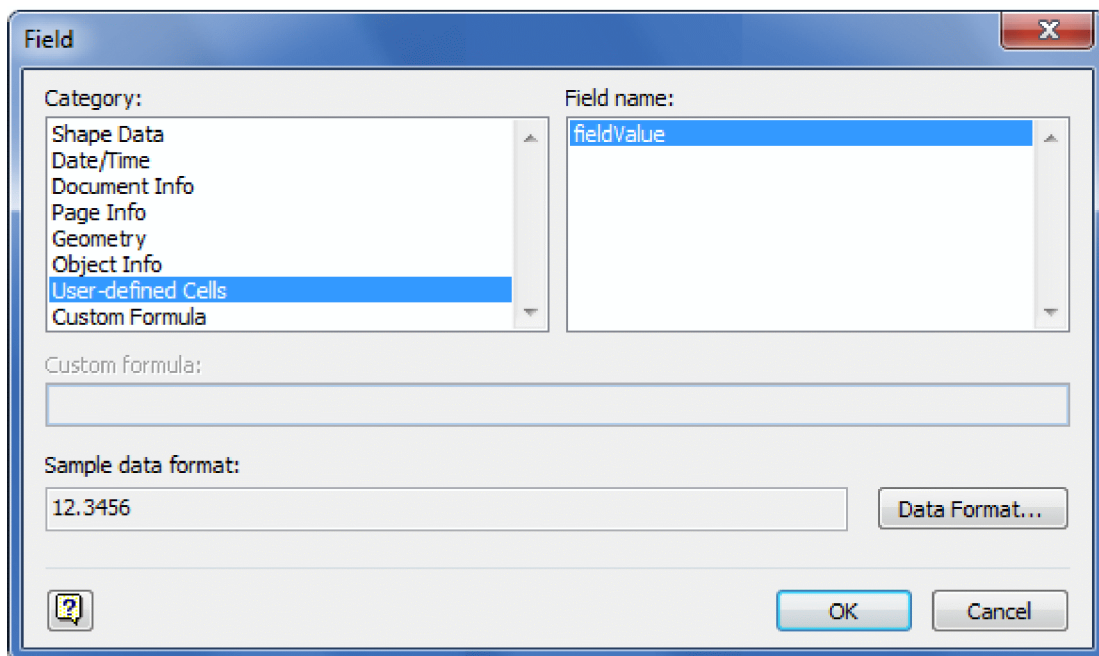
User-defined Cells	Value	Prompt
User.fieldValue	Sheet.1!Prop.Customer	""

User-defined Cells	Value	Prompt
User.fieldValue	Sheet.1!Prop.JobNo	""

User-defined Cells	Value	Prompt
User.fieldValue	Sheet.1!Prop.SubJobNo	""

Now that we've created a User-defined entity for each of the sub-shapes, it's time to insert the group (parent) shape data as fields into their respective sub-shape text blocks. I will show you how to do it for the 'DRAWING NO:' sub-shape's text block and you can repeat it for the others:

1. Close the Shapesheet window and select the 'DRAWING NO:' sub-shape. To enter into edit mode press F2, then place the cursor at the tab spacing where the Prop.DrawingNo text will be inserted.
2. Go to [Insert](#) → [Fields...](#) and you'll see fieldValue under User-defined Cells. Click OK.



Repeat the two step process for the other sub-shapes. You may want to include the sub-shape above the REV column in the amendments (document revisions) text blocks to share the same revision number as the REV: sub-shape at the lower right of the main title block, since it will be the final last edited 'Original' version to be submitted for approval.

The completed main title block template will look like the figure below. You can select and directly type into the sub-shapes above the AMENDMENTS and APPROVED columns per se.

0	2015-01-06		
REV	DATE	AMENDMENTS	APPROVED
DRAWING TITLE : TRACKING RADAR INTERFACE BOARD			
CUSTOMER : INTELLi-SYSTEMS		 Kinetec Corp. A company of Associated Engineers in Electronics	
DRAWN BY: NG KENG TIONG		JOB NO: 2014-12-0386	SUB-J/NO: 0386-001
REVIEW BY: RANDY DYKEMAN		DRAWING NO: LD123456-050	REV: 0 SHEET: 1 OF 4

Figure 5.14 The final completed main title block template.

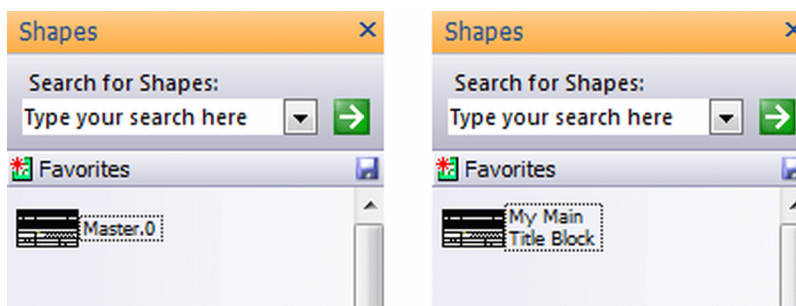
Lastly, we want to add this main title block template to our Favorite stencil so we can drag and drop it into the background of new drawings in the future. Before we do that, let's erase the Shape Data from our completed title block template:

1. Select the main title block and bring up its Shapessheet.
2. Click on each of the Value cell in the Shape Data section and type "" (no white spaces) followed by the ENTER key.
3. As you do so, the inserted fields for DRAWING TITLE:, REV:, CUSTOMER:, JOB NO: and SUB-J/NO: will become blank.
4. Leave the rest of the sub-shapes with fields that are filled as these will be automatically updated when you drag and drop the title block shape from the stencil onto a new drawing.

Shape Data	Label	Prompt	Type	Format	Value	SortKey	Invisible	Ask
Prop.DrawingNo	"DRAWING NO"	No Formula	0	No Formula	""	No Formula	No Formula	TRUE
Prop.Revision	"REVISION"	No Formula	0	No Formula	""	No Formula	No Formula	TRUE
Prop.Customer	"CUSTOMER"	No Formula	0	No Formula	""	No Formula	No Formula	TRUE
Prop.JobNo	"JOB NO"	No Formula	0	No Formula	""	No Formula	No Formula	TRUE
Prop.SubJobNo	"SUB-JOB NO"	No Formula	0	No Formula	""	No Formula	No Formula	TRUE

To add the title block shape to the Favorite stencil:

1. Go to **File** → **Shapes** → **My Shapes** → **Favorites**. The Shapes side bar will appear on the left of the drawing page with the blank Favorites stencil.
2. If this is the first time you're opening, the stencil will be read-only; to make it editable for you to add new master shapes, right click on the Favorites title bar and select Edit Stencil.
3. Now go back to the drawing page and select our main title block, then drag and drop it into any blank area of the Favorites stencil. The main title block will disappear from the drawing page and a new master shape with the name Master.0 will be added to Favorites.
4. You can rename the master shape to 'My Main Title Block' or some other name.



Next time you open a new drawing page for a project with different file properties information, and then open the Favorites stencil to drag and drop an instance of 'My Main Title Block' into a background page, you will see fields updated automatically and be prompted to enter the necessary shape data:

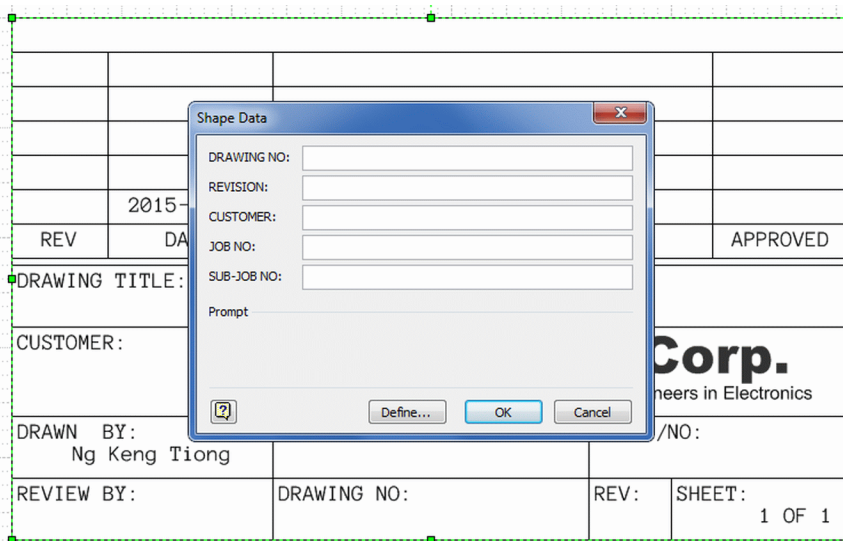


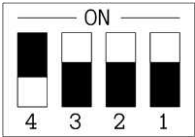
Figure 5.15 New title block with automatic updating and shape data entry.

And that completes our multi-field tutorial. (Whew!)

A Multi-Shape (DIP-Switch) Example

If you thought multi-field shapes are fun after going through the last exercise, wait till you try your hands on multi-shape! In some ways they are like identical twins, since they have the ability to effect changes on fields, shapes, or both. What set them apart are the functional aspects for which they are created—text versus geometry. We have already seen how text in shapes can be used in a powerful way when grouped into a single entity. We shall now look at how geometry in shapes can be manipulated to give that extra 'wow factor' when you put them together.

The candidate we want to consider this time is the 4-way DIP switch which we've created in [Chapter 3](#) for the SCSI host adapter card, a multi-shape that is formed by grouping five sub-grouped shapes, i.e. a mechanical body (a rectangle, a line, and five text boxes) and four individual slide switches (a white square and a black rectangle).



If the DIP switch is a static (dumb) shape, in order to change the position of the switch bits, you'll need to first ungroup it, select the switch bits to be changed and perform a vertical flip operation, then do a re-group of the shapes again. If this is the only DIP switch in the PCB layout, it's not a problem, though it might be a hassle if you need to do it a couple of times for different configurations of the PCB. What if you have a number of these DIP switches on a single PCB? Wouldn't it be easier and quicker to just bring up the shape data dialog box, enter a number representing the switch bits' positions and see the changes effected straightaway? Sounds good to me, so let's do it!

Assuming that the DIP switch is the only shape on the drawing page, the group (parent) shape should have an ID of 1 and any reference made to it from any of the sub-shapes will be 'Sheet.1' since it is the top most level. The rest of the sub-shapes will have different IDs depending on the order in which they're created, but we need not be concerned with them. Let's add a shape data to the DIP switch:

- 1. Right click on the shape and select [Data](#) → [Shape Data...](#)
- 2. Click [Yes](#) to bring up the Define Shape Data dialog box.
- 3. Enter the following information:

Label : SETTING
Type : Number (choose from the drop down menu)
Prompt : Enter a number between 0-15¹⁹⁶

Click OK.

You can key in any number (say, 7) into the SETTING field in the Shape Data dialog box but nothing will happen because we have not link the shape data we just defined with the switch bits sub-shapes yet. If you open the group (parent) shape's Shapesheet, this is what you'll see under the Shape Data section, after you've rename Prop.Row_1 to Prop.Setting:

Shape Data	Label	Prompt	Type	Format	Value
Prop.Setting	SETTING	Enter a number between 0-15	2	No Formula	7.0000

¹⁹⁶ Four binary positions equal 16 possible combinations (hope you still remember your basic digital numbers).

What we want to do next is to evaluate the shape data's value (Prop.Setting) and effect a vertical flip on the switch bits that need to be changed. Each of the switch bits must have its Shape Transform section individually edited, so we'll start with the lowest bit '1' switch:

1. Select switch bit 1's sub-shape and bring up its Shapessheet:

Shape Transform					
Width	0.1250 in	PinX	0.6875 in	FlipX	FALSE
Height	0.3125 in	PinY	0.2813 in	FlipY	FALSE
Angle	0.0000 deg	LocPinX	0.0625 in	ResizeMode	0
		LocPinY	0.1563 in		

The FlipX and FlipY properties are by default FALSE because after the shape has been created there is no flip horizontal or vertical operations performed on it yet.

2. We want to effect a vertical flip on the switch bit sub-shape so click on the FlipY's Boolean field (FALSE, as above) and type the following line into the Shapessheet formula bar:

```
IF(BITAND(Sheet.1!Prop.Setting,1),TRUE,FALSE)
```

BITAND is a built-in function that can test and change the property of a shape that is stored as a bitmask. In the case of switch bit 1, we want to test if the first bit of the Prop.Setting value of the group (parent) shape is 1—if it is then FlipY will be set to TRUE, else it will be set to FALSE. Since Prop.Setting has a value of '7' as seen earlier, bit 1 is indeed a '1' so once you entered the above statement into the Shapessheet formula bar and click on the green tick to accept, FlipY will immediate change to TRUE and the switch bit 1 sub-shape will perform a vertical flip at the same time!

Repeat steps 1-2 for the other three switch bits sub-shapes, and type the following lines into their respective Shapessheet formula bar for the FlipY property's Boolean field:

```
Switch bit 2  IF(BITAND(Sheet.1!Prop.Setting,2),TRUE,FALSE)
Switch bit 3  IF(BITAND(Sheet.1!Prop.Setting,4),TRUE,FALSE)
Switch bit 4  IF(BITAND(Sheet.1!Prop.Setting,8),TRUE,FALSE)
```

Notice how the switch bits sub-shapes react to the result of the BITAND evaluations each time you accept the formula statement you typed into the FlipY Boolean field. The test values of 2, 4 and 8 are binary bit positions for switch bits 2-4, respectively:

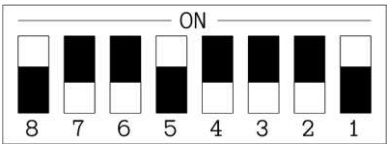
```
Switch bit 1  1 = 0001          Switch bit 3  4 = 0100
Switch bit 2  2 = 0010          Switch bit 4  8 = 1000
```

We're done with the 4-bit DIP switch example. Now if you bring up the Shape Data dialog box and type in a value between 0-15, the switch bits sub-shapes will flip positions to reflect the changes accordingly. But wait, we're not done yet...

Doing binary math for a 4-bit DIP switch is a piece of cake. To change the positions to binary 0101 or 1010, we can easily deduce the decimal values as 5 or 10. But what if we're dealing with an 8-bit DIP switch? Try deducing the decimal value for 10110010! It's going to take you a while to figure out that it's decimal 178 and if you have a couple of 8-bit DIP switches to handle or thinking to change the settings of a single 8-bit DIP switch multiple times—I think you get the idea...

Wouldn't it be a breeze if you could just type in the setting values as they are, in binary format, instead of taking pains to convert them to decimal numbers? Well, be careful what you wish for because you're just about to have your wish granted!

An 8-bit DIP switch has twice the number of switch bits compared to a 4-bit DIP switch, but in terms of binary combinations it has not twice but sixteen times the amount compared to the latter, with a value range from 0-255! Now you know why it doesn't make sense to use the same approach to create an 8-bit DIP switch the way you do with a 4-bit one. It's just not a smart thing to do (pun intended).



Again, assuming that the 8-bit DIP switch is the only shape on the drawing page, but this time formed by grouping nine sub-grouped shapes, i.e. a mechanical body (a rectangle, a line, and five text boxes) and eight individual slide switches (a white square and a black rectangle). Let's add a shape data to the DIP switch:

1. Right click on the shape and select **Data** → **Shape Data...** and click **Yes** to bring up the Define Shape Data dialog box.
2. Enter the following information:
Label : SETTING
Type : Number (choose from the drop down menu)
Prompt : Enter an 8-bit binary number (e.g. 00110110)¹⁹⁷
Click OK.
Leave the SETTING field empty in the Shape Data dialog box and click OK again.

Now open its Shapessheet and rename Prop.Row_1 to Prop.Setting under the Shape Data section to get the following:

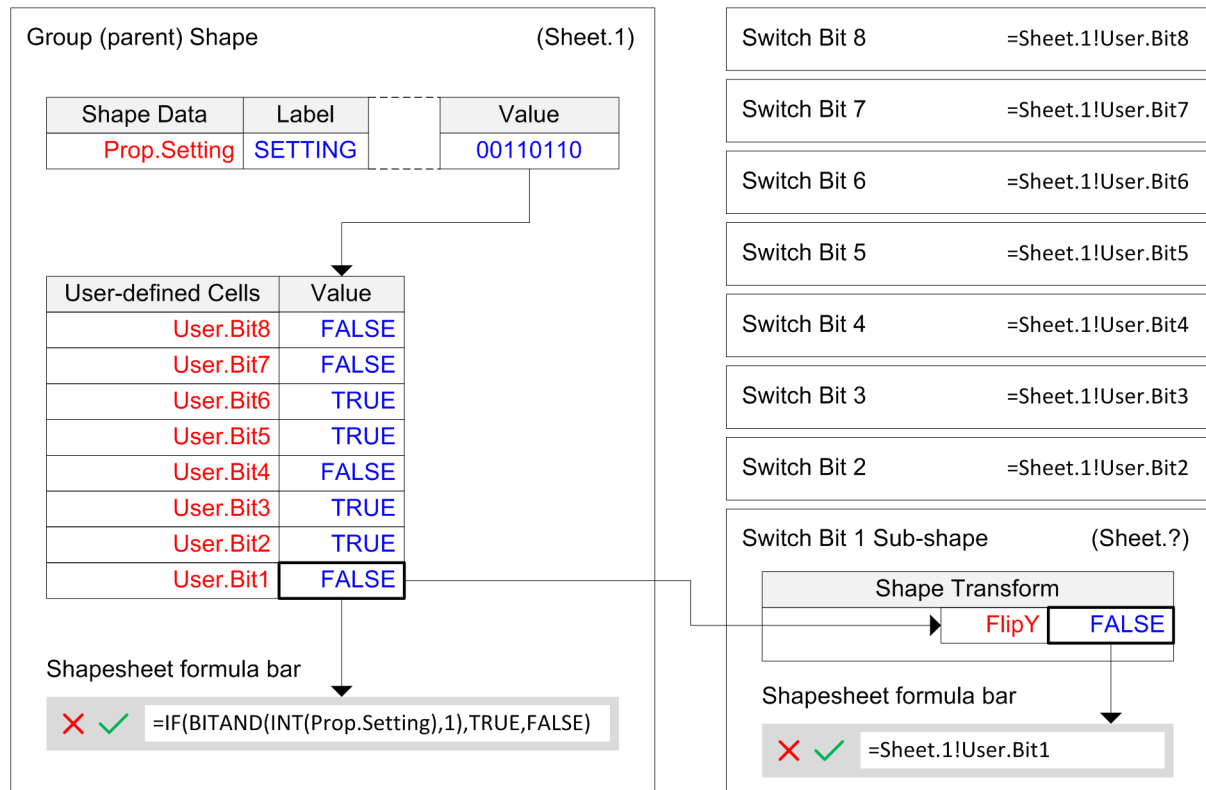
Shape Data	Label	Prompt	Type	Format	Value
Prop.Setting	SETTING	Enter an 8-bit binary number (e.g. 00110110)	2	No Formula	No Formula

Next, we want to evaluate the shape data's value (Prop.Setting) and effect a vertical flip on the switch bits that need to be changed. This time, however, we need to take a different approach because we're not entering a decimal number but an 8-bit binary number.¹⁹⁸

¹⁹⁷ Eight binary positions equal 256 possible combinations, i.e. 2 to the power of 8 (2⁸).

¹⁹⁸ Well, actually we ARE still entering a decimal number, but we are giving the user the illusion that it's a binary number with the prompt message. Tricky, aren't we? (snigger...)

Instead of going to each switch bit sub-shape's Shapesheet to edit the FlipY Boolean field, which can be a rather tedious thing to do if you need to experiment, debug and make changes—especially if you're doing for eight sub-shapes here—it would be better to do the formulas on the group (parent) shape's Shapesheet by means of User-defined cells and then referencing the results from the sub-shapes using the fixed user-defined names, as shown below:



Here's how create the User-defined Cells section:

1. Select the group (parent) shape and bring up its Shapesheet.
2. Right click on any area of the Shapesheet and choose Insert Section... then select User-defined cells and click OK. A User-Defined Cells section with a single row named User.Row_1 appears.
3. Right click on that row and select Insert Row. User.Row_2 appears below.
4. Press F4 six times to insert another six rows to make the total eight.

The values in these user-defined cells will initially be assigned to 0, but will change based on the type of formulas that we put in later. We want to change the names to reflect the switch bit positions, so click on the first row's name field and type 'Bit8'. As usual, don't bother to include the prefix 'User.' because it will automatically be added to the names you assigned. Do this for the remaining rows in descending order so you get what is shown in the figure above.

Chapter 5

Next, select each individual user-defined cell's value field and type the formula listed below into their respective Shapessheet formula bar and click the green tick to accept (or press the ENTER key):

User.Bit8	IF(BITAND(INT(Prop.Setting/1E7),1),TRUE,FALSE)
User.Bit7	IF(BITAND(INT(Prop.Setting/1E6),1),TRUE,FALSE)
User.Bit6	IF(BITAND(INT(Prop.Setting/1E5),1),TRUE,FALSE)
User.Bit5	IF(BITAND(INT(Prop.Setting/1E4),1),TRUE,FALSE)
User.Bit4	IF(BITAND(INT(Prop.Setting/1E3),1),TRUE,FALSE)
User.Bit3	IF(BITAND(INT(Prop.Setting/1E2),1),TRUE,FALSE)
User.Bit2	IF(BITAND(INT(Prop.Setting/1E1),1),TRUE,FALSE)
User.Bit1	IF(BITAND(INT(Prop.Setting),1),TRUE,FALSE)

I've chosen to use the exponential format for the divisor values instead of typing zeros for clarity, as well as to let you know it's possible to include such numerical format which Visio accepts and subsequently converts back to decimal numbers (10, 100, 1000...) when you press the ENTER key. This is less error prone than trying to count the number of zeros for the larger numbers.

The last step is to go to the individual switch bit sub-shape's Shapessheet, select the FlipY field under the Shape Transform section and type in the 'User.Bit?' name of the user-defined cell prefixed with the group (parent) shape ID (Sheet.1!) as follows:

Switch Bit 1	Sheet.1!User.Bit1	Switch Bit 5	Sheet.1!User.Bit5
Switch Bit 2	Sheet.1!User.Bit2	Switch Bit 6	Sheet.1!User.Bit6
Switch Bit 3	Sheet.1!User.Bit3	Switch Bit 7	Sheet.1!User.Bit7
Switch Bit 4	Sheet.1!User.Bit4	Switch Bit 8	Sheet.1!User.Bit8

We're done!

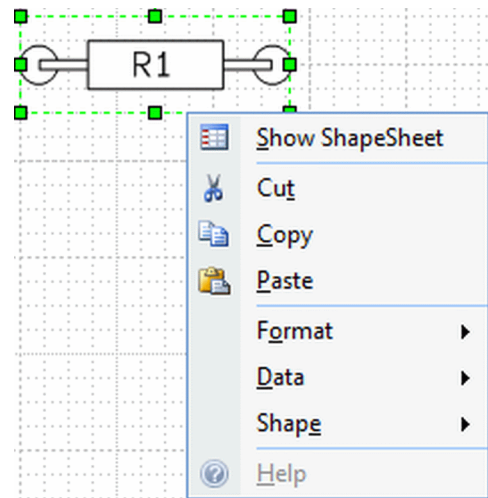
Now if you right click on the 8-bit DIP switch and select Data → Shape Data... to bring up the Shape Data dialog box, then type in a 'binary' number (e.g. 10110010), you'll see the DIP switch's individual bits flip accordingly based on the evaluated result from your input. No need for any tedious computation of the binary value into decimal equivalent—simple and elegant!

Oh, before I forget, there's one more thing about smartshapes that I think would be useful to know... (don't worry, it's easy as ABC!)

Adding a Shape Context Menu

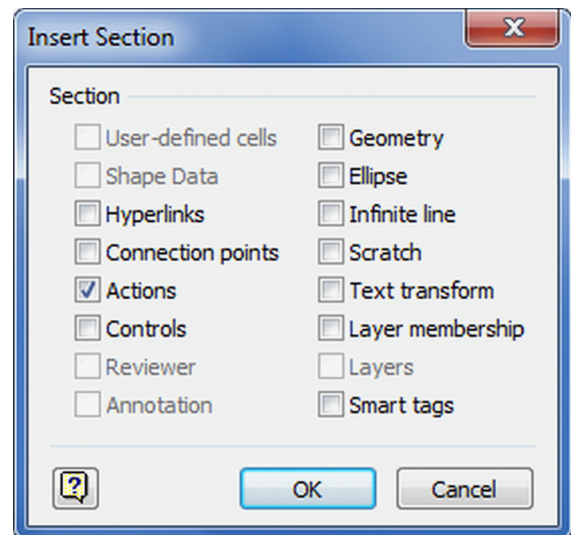
A shape context menu is a menu that shows up when you right click on a page or shape in Visio, giving you that added option to perform certain related operations pertaining to the selected object. The [Show Shapessheet](#) (single level) and [Data](#) → [Shape Data...](#) (multi-level) are some examples that that we've used so far (see right figure).¹⁹⁹

But do you know that you can actually add your own custom context menu items to a shape via its Shapessheet to give it extra functionalities (e.g. Rotate Right or Left) for users to choose from directly, instead of reaching out to the Action Toolbar to perform those operations or navigating multiple levels in the menu to look for a certain shape manipulation function to accomplish the task?



Let's do a simple exercise by taking our resistor layout symbol and give it a context menu option to rotate left or right. Here are the steps:

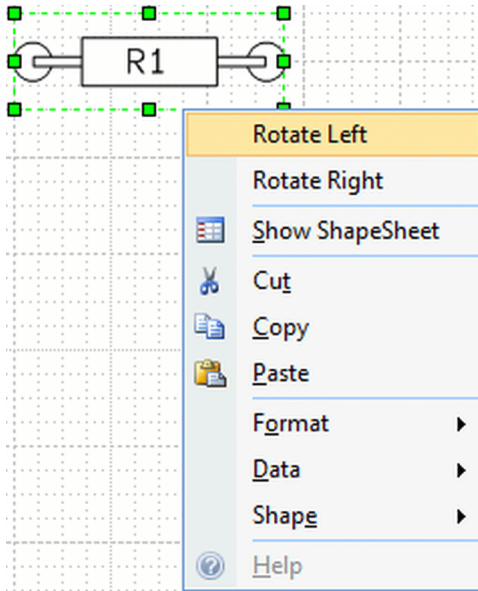
1. Select the resistor layout shape and bring up its Shapessheet.
2. Right click anywhere in the Shapessheet and select [Insert](#) → [Section...](#) then tick on the [Actions](#) checkbox and click [OK](#).
3. Go to the Actions section, right click on any cell and select [Insert Row](#). Now you have two rows named [Actions.Row_1](#), [Actions.Row_2](#). There is no need to rename them since we're just doing a simple demo.
4. Type the following statements into the Action and Menu fields for both [Row_1](#) and [Row_2](#) as shown below:



Actions	Action	Menu
Actions.Row_1	SETF(GetRef(Angle),Angle+90 deg)	"Rotate Left"
Actions.Row_2	SETF(GetRef(Angle),Angle-90 deg)	"Rotate Right"

Close the Shapessheet and you're done!

¹⁹⁹ Visio 2007 and the earlier version only support single level context menu, while Visio 2010 and later are able to support multi-level like the Data's grouped items. So if you want this feature, then go for Visio 2010 or later!



Now when you select the resistor layout symbol and right click to bring up its context menu, you'll see the two added custom menu items (Rotate Left, Rotate Right) for you to select which, when you do, will produce the Shape Transform operation just as you would see when you click on the Rotate Left and Rotate Right buttons on the Action Toolbar.

You can certainly change the degree of rotation to any value other than 90 degrees, but for the resistor layout symbol, I don't see the logic for odd angles of rotation other than the usual 90-degree!

If you want to learn more about what you can create with the custom context menu ability of Visio, I encourage you to look up the references on this topic which you can easily find online and in many Visio books mentioned in the Summary on Shapes below.

Summary on Shapes

This section has given you a foretaste of what Visio shapes can do, covering the fundamentals of shape data, shape (data) graphics and shape sheets, while illustrating basic to pre-intermediate concepts in electronics illustration applications. There are certainly alternative ways to go about doing it, from the crude and unorganized to the more refined and complicated; however, the key to a successful and satisfying drawing experience is to find the right mix of techniques, and in the process achieve a balance between tapping the full potential of Visio while keeping things simple and within manageable levels.

If you're interested to learn more about Visio shapes, there are a number of good books and online resources that can bring you up to speed:²⁰⁰

- Visualizing Information with Microsoft Office Visio 2007 by David Parker
- Using Microsoft Visio 2010 by Chris Roth*
- Visio 2007 on MSDN
- Developing Microsoft Visio Solutions by Microsoft Press

* Alternatively, you can visit his website at www.visguy.com where he writes many insightful and useful articles on doing stuff in Visio. He'll usually reply comments and questions to his posts so you can try asking him questions if you do encounter problems related to Visio—he's the Visio guy, after all!

²⁰⁰ Hopefully, this book will achieve a good measure of success to enable me to quit my job so I can focus full-time on writing more technical books based on my years of engineering experience, one of which will be on advanced Visio graphics for engineering illustrations.

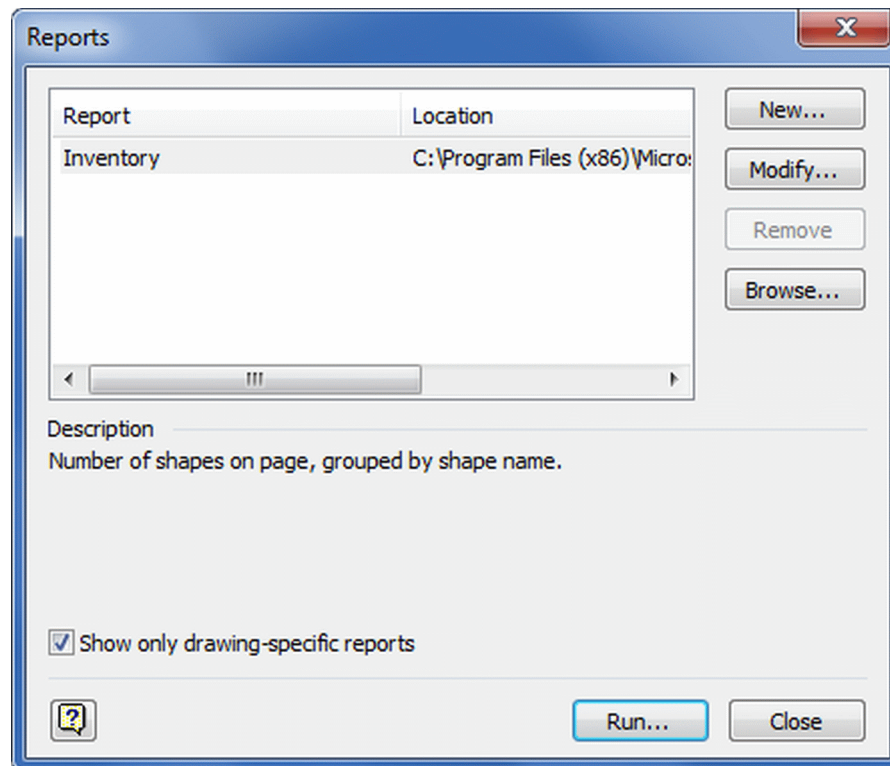
GENERATING BILL OF MATERIALS

In [Chapter 2](#) I outlined the need to create a bill of materials (BOM) as one of the basic preparation tasks before you commence reverse-engineering a PCB. Once you populated the layout diagram and added in the shape data for the various component symbols, you can use Visio to generate a report summarizing those data and export it to an external format such as Excel, Html, XML or even a Visio Shape!²⁰¹

Let's use the simple SCSI host adapter card we created in [Chapter 3](#) for illustration. I have already added the necessary shape data (REFDES, PARTNO, and DESC) to the component symbols and I assume by now you should have no problem doing that, so we will not go into the details.

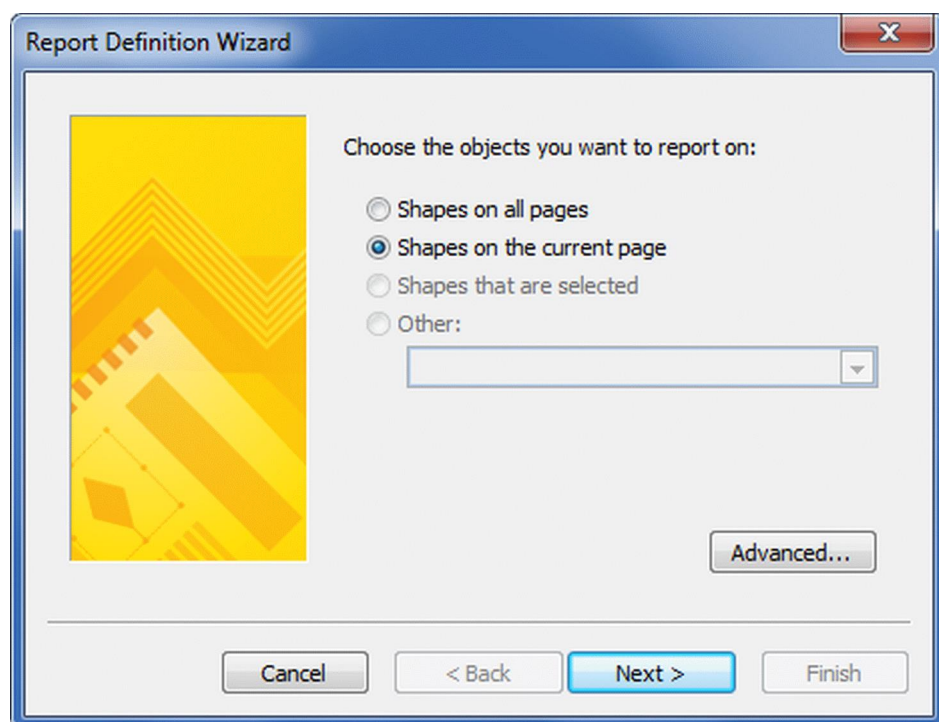
To generate a bill of materials (BOM) report using the shape data:

1. Make sure you're on the drawing page containing the SCSI host adapter card layout diagram. Go to [Data](#) → [Reports...](#) The Reports dialog box appears:

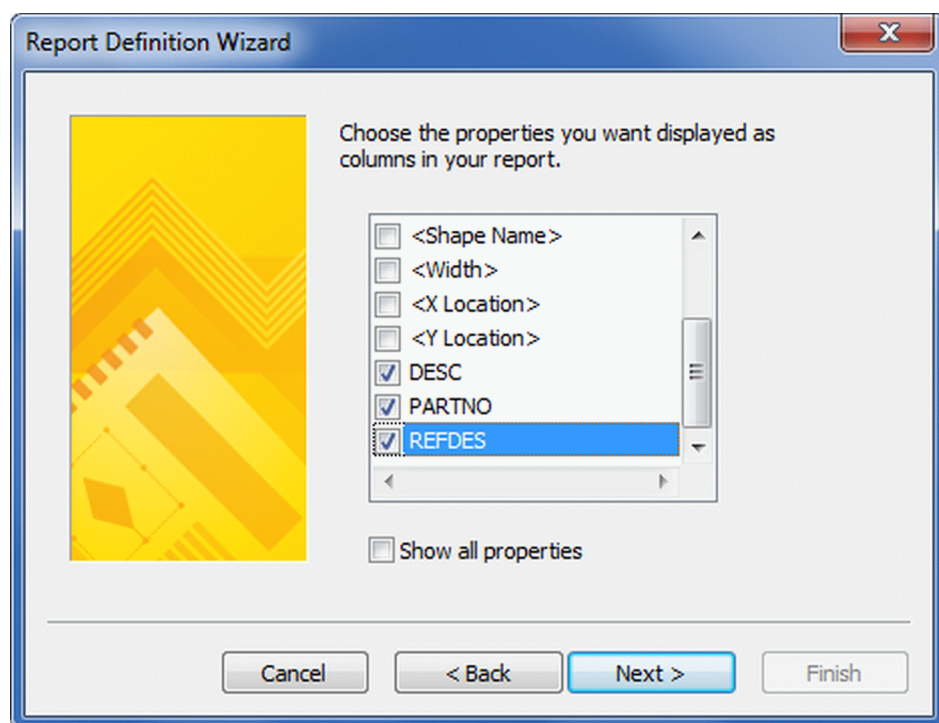


We want to create a new report with fields based on the shape data, so click on [New...](#)

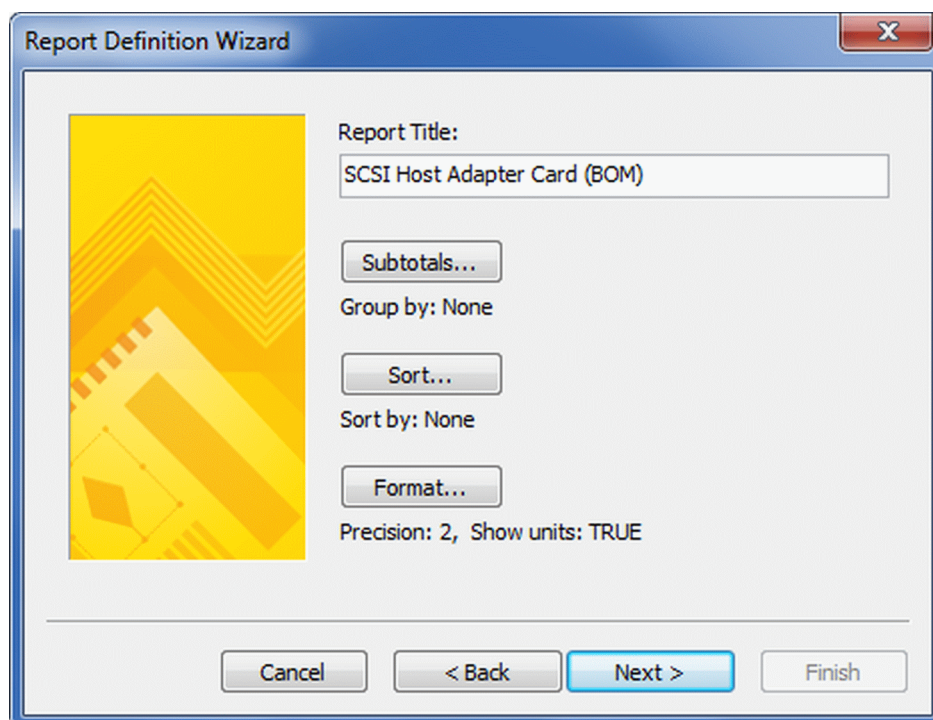
²⁰¹ You may be wondering if you should create a BOM in the first place during the basic preparation phase, since Visio is able to generate one. Bear in mind that you still need to take an inventory of the components on the PCB you're working on to gather their datasheets in order to be able to create their layout symbols. I did emphasize to keep that preliminary BOM as simple as possible, including just the REFDES and PARTNO of the components to assist you to keep track of your progress on data collection.



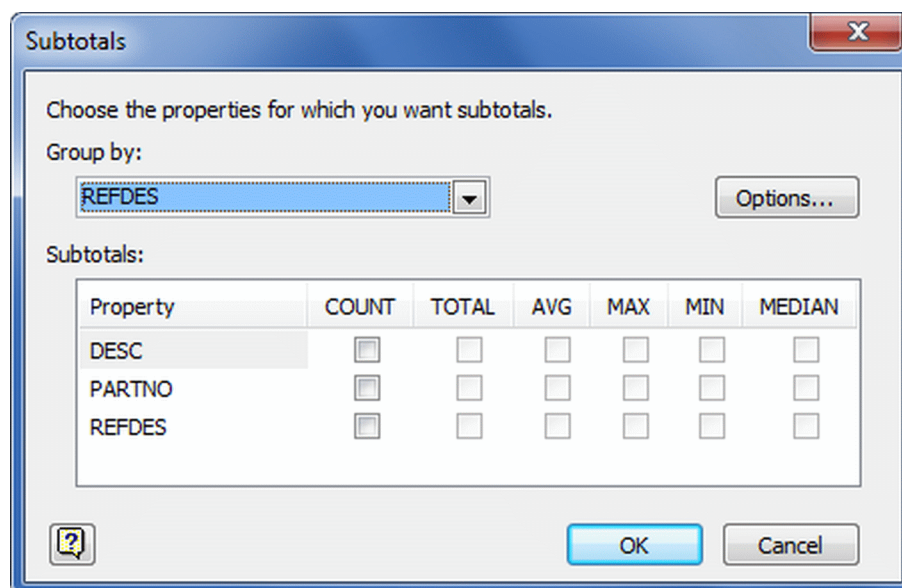
2. The Report Definition Wizard appears and will guide you through the process to create a new report format. If your drawing spans multiple pages, select 'Shapes on all pages'. In our case, we only have one drawing page which is the current page, so just click [Next](#).



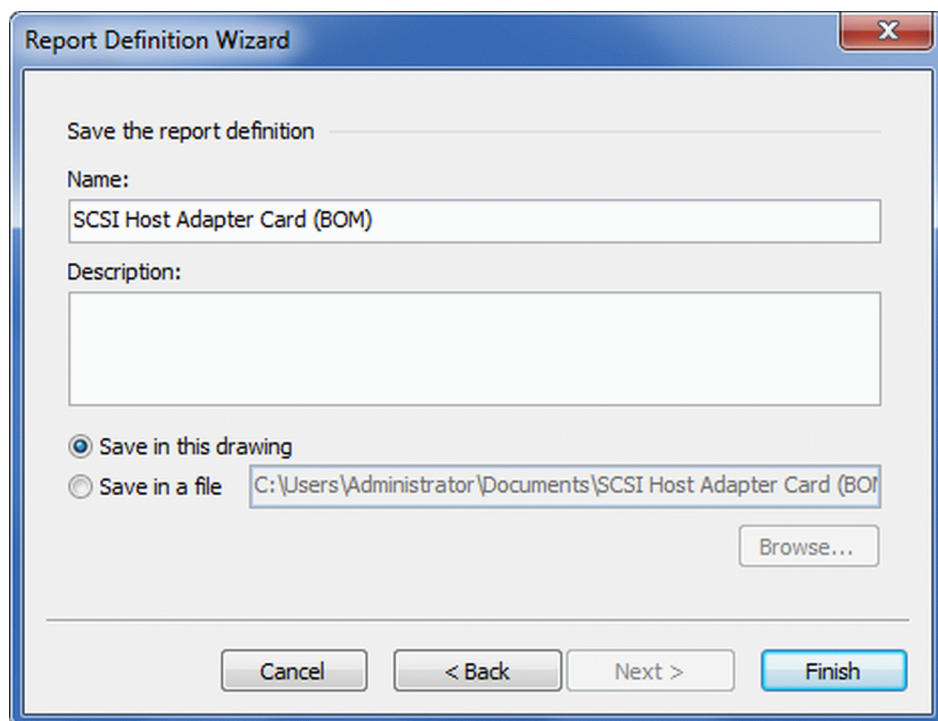
3. Scroll down the list and select the data (DESC, PARTNO, REFDES) you want to be included in the report, then click [Next](#).



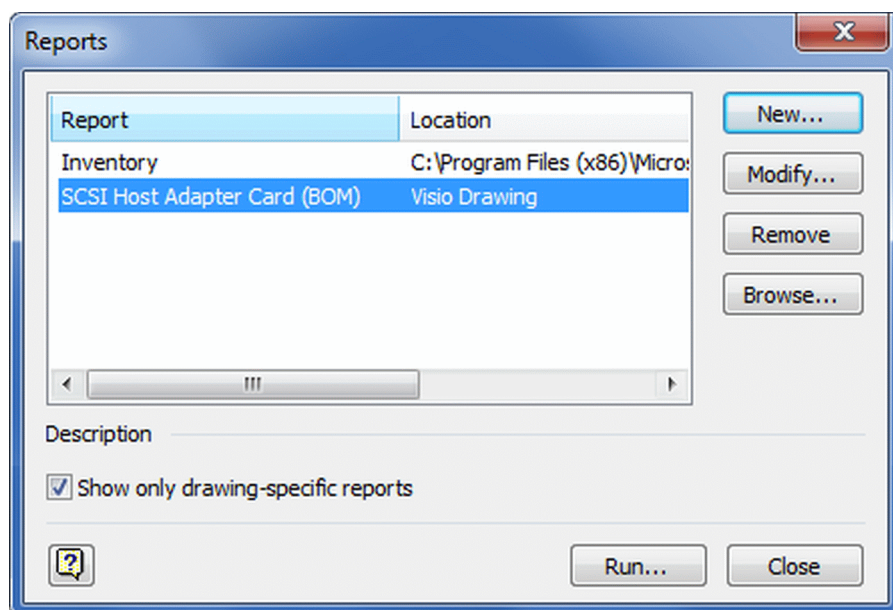
4. Give the report a title (e.g. SCSI Host Adapter Card (BOM)), then click [Subtotals...](#) and select REFDES in the [Group by:](#) drop down menu. Click [OK](#) follow by click [Next](#) again.



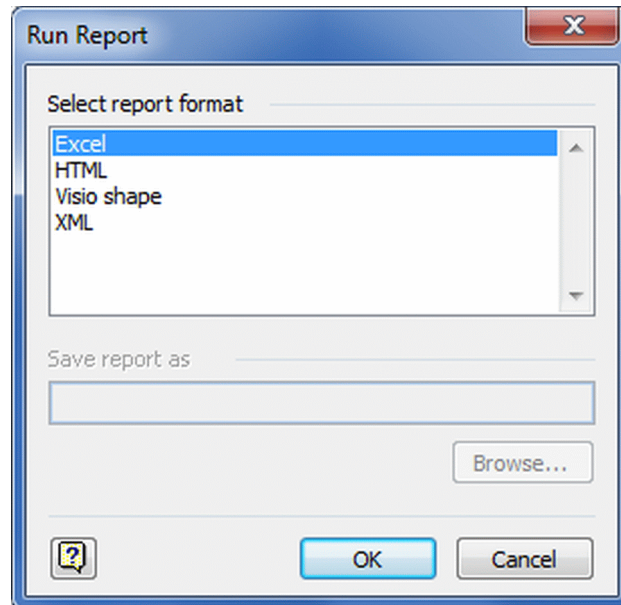
5. Give the report definition the same name as the report title (or a different name if you prefer) and leave the option as 'Save in this drawing' for the .vrd file instead of saving the Visio report definition file externally. Click [Finish](#).



6. The Reports dialog box's Report list now contain the new report entry for the SCSI host adapter card and its location is indicated as inside the Visio drawing. Click [Run...](#) to generate the report.



7. The Run Report dialog box appears. We want to export the shape data to Excel so accept the default selection and click OK.



After some processing, the Excel application will open with the three columns of REFDES, PARTNO and DESC fields and their data, together with the report title as shown in Table 5-3 overleaf.²⁰²

Recall on page 200, I mentioned that the numbers depicted in the [Layer Properties](#) dialog box is much larger than the actual component count. The reason for this is because Visio keeps track of objects on each layer based on individual shape basis and not groupings. So if you create an IC layout symbol with ten shapes grouped together as an IC object, Visio will report it as ten and not one! It is not necessary to insist on making the [Layer Properties](#) dialog box reflect the correct component count, though you could do so using [Shape](#) → [Operations](#) → [Combine](#) method of grouping shapes but there are complications which you'll have to manage, including the penalty of loss of individual shapes as they are combined or merged, which means you'll not be able to modify parts of the combined object any more.

Well, we are done with the advanced stuff involving Visio! Before we round up and close this Chapter, I thought it would be beneficial to touch on something more electronic in nature and related to the practical aspect of reverse-engineering. Get ready to be blown away by...

²⁰² Note that Excel sorts alphanumeric data in a rather odd manner (refer capacitor and IC REFDES fields) that has earned the irks of many users, but this can be circumvented using some Excel tweaks that you can easily find in online forums.

Table 5-3 Excel report (BOM) generated by Visio

SCSI Host Adapter Card (BOM)		
REFDES	PARTNO	DESC
C1	C410C104Z5U5CA	Capacitor, 0.1uF 50V Ceramic Axial Lead
C10	C410C104Z5U5CA	Capacitor, 0.1uF 50V Ceramic Axial Lead
C11	C410C104Z5U5CA	Capacitor, 0.1uF 50V Ceramic Axial Lead
C12	CX05D476K	Capacitor, 47uF 6V Tantalum Axial Lead
C13	CX05D476K	Capacitor, 47uF 6V Tantalum Axial Lead
C2	C410C104Z5U5CA	Capacitor, 0.1uF 50V Ceramic Axial Lead
C3	CX05D12N6K	Capacitor, 12.6nF 6V Tantalytic
C4	C410C104Z5U5CA	Capacitor, 0.1uF 50V Ceramic Axial Lead
C5	C410C104Z5U5CA	Capacitor, 0.1uF 50V Ceramic Axial Lead
C6	C410C104Z5U5CA	Capacitor, 0.1uF 50V Ceramic Axial Lead
C7	C410C104Z5U5CA	Capacitor, 0.1uF 50V Ceramic Axial Lead
C8	MON104-X7R-50B	Capacitor, 100nF 50V Ceramic Radial Lead
C9	C410C104Z5U5CA	Capacitor, 0.1uF 50V Ceramic Axial Lead
CR1	1N5817	Diode, Rectifier 1A Schottky Barrier
F1	0267-01.5	Fuse, 1.5A 125V Very Fast Acting
J1	2213S-50G	Header, 50-Way 2-Row Vertical
J2	57-BC50B-AM100	Connector, Female 50-Way Centronics
R1	MRA0207-1MBTA15	Resistor, 1K Precision Metal Film
RP1	007-6718203	Resistor Network, 220/330 Dual Terminator 8-SIP
RP2	007-6718203	Resistor Network, 220/330 Dual Terminator 8-SIP
RP3	007-6718203	Resistor Network, 220/330 Dual Terminator 8-SIP
U1	SN74LS688N	IC, 8-Bit Magnitude Comparator
U10	74LS244N	IC, Octal Buffers and Line Drivers
U2	76YY21514S	Switch, 4-Way DIP
U3	SN74LS04N	IC, Hex Inverters
U4	DM74LS138N	IC, 3-to-8 Line Decoder
U5	SP8924	IC, Custom Part (NCR)
U6	SP8918	IC, Custom Part (NCR)
U7	NCR-5380	IC, SCSI Interface Device
U8	D2764A	IC, 8K x 8 EPROM
U9	SN74LS688N	IC, 8-Bit Magnitude Comparator

Note: This report has been formatted from the plain text output generated by Visio.

THE MATRIX

"Unfortunately, no one can be told what the Matrix is. You have to see it for yourself."²⁰³

Well, I'm not sure if Neo understood Morpheus when he was first introduced to a subject as abstract and intriguing as the Matrix itself. Fortunately, the subject of programmable logic devices (PLDs) is not as difficult to comprehend for anyone with half an engineer's mind. Nowadays, it's common to find these devices on PCBs so if you're going to do serious reverse-engineering work, you had better get equipped with some knowledge on what PLDs are and why they have become such an indispensable and vital part of modern circuit designs.²⁰⁴

I won't be going into too much details on PLD basic theory or giving introductory tutorials on these devices—you can find plenty of them online and in prints already. I would recommend Clive Maxfield's *Bebop to the Boolean Boogie* for an unconventional guide to electronics; besides loads of good stuff and his humorous but engaging style of writing, there's a chapter on programmable ICs that is quite well explained it'll clear up many questions and doubts on this topic. He has also written a number of books on the more complex FPGA chip, which is beyond the scope of my discussion here. In fact, just dealing with the standard PLD architecture alone is a big undertaking, considering there are diverse structures and logic implementations like the fuse-link PLAs and PALs, reprogrammable PLDs such as GALs,²⁰⁵ and the more complex PLDs (CPLDs).

So go ahead, take the **red** pill...²⁰⁶

²⁰³ This is Morpheus' famous line to Neo in the 1999 blockbuster, *The Matrix*, which opened up a new paradigm in the way we view and appreciate sci-fi movies, setting an exciting trend for future film-making of this genre that engages the mind and perception on a whole new level.

²⁰⁴ Twenty years ago, there was this colleague of mine, a petite lady engineer, who worked on developing a test program for a digital PCB containing a whopping twenty-five PLDs, registered and non-registered types. As the PLD design files containing the device equations were not available, she had to individually remove the ICs and read out the fuse map content in JEDEC format on a Data I/O programmer. After that, she had to painstakingly transfer the data to the respective PLD's logic diagram templates, and then decipher the AND-OR connectivity to obtain the Boolean logic correlation between the inputs and outputs, in order to be able to write the in-circuit test routines (ICTRs) to test each of these ICs. It was such a mentally grilling and time-consuming process that at the end of it, she suffered a severe migraine headache and had to take three days' medical leave! Thankfully, recognizing this problem the management finally invested about \$30,000 on a perpetual license for an ATG (automatic test generation) software that produces test vectors from the JEDEC data, which not only cut down test development time substantially but also saved us all from possible mental breakdown as well.

²⁰⁵ These AND-OR array-based PLDs are also known as Simple PLDs (SPLDs) as opposed to the more complex CPLDs using macrocell-based architectures. Of course, an SPLD does contain a small number of macrocells to be exact—I'm simplifying just for the sake of comparison between the two here.

²⁰⁶ "This is your last chance. After this, there is no turning back. You take the blue pill—the story ends, you wake up in your bed and believe whatever you want to believe. You take the red pill—you stay in Wonderland and I show you how deep the rabbit-hole goes." (Morpheus' irresistible offer to Neo as he placed the two pills right in front of him.)

JEDEC and Fuse Map

First thing first. You have a PCB on hand containing one or more PLDs and you're wondering whether it's worth the trouble to perform the extra steps outlined below to obtain the device equations, so you do not waste time validating any unused input or output pins. Of course, you can go the brute-force way and treat the PLDs (if any) as black boxes until you reach a reasonable conclusion through repeated validations, or you could invest a little more effort upfront and identify unused pins to remove guess work.

Since you've taken the red pill, I assume you must have opted for the latter. But before we begin, I need to come clean with you on a few points. The approach I am about to discuss hinges on the following assumptions:

1. The PLD is of the SPLD type and must not be security-bit protected, that is, you must be able to read out its fuse map content into a JEDEC file,
2. You should have some basic knowledge about PLDs—preferably have worked on some kind of EPROM/PLD programmer, knows what a JEDEC file is and how to read out a PLD's content and save it for viewing later,²⁰⁷
3. You still remember your Boolean algebra and can understand basic equations of this sort, and
4. You are comfortable with simple MS-DOS command prompt operations.

I'll use the PAL16L8 chip for my illustration and example. Consider the fuse map of the JEDEC file²⁰⁸ shown overleaf. There are two formats defined in the JEDEC standards, which are listed in the Data I/O 3980xpi programmer's User Manual²⁰⁹ as JEDEC full format (code 91) and JEDEC kernel mode (code 92). The fuse map shown is of the latter, as evidenced from the starting fuse element number in each row beginning with `*Lnnnnn` followed by a 32-bit string of 1s and 0s, 1 depicting a high-resistance link (open fuse) and 0 a low-resistance link (intact fuse). I've also colored the rows in groups of eight for easier reference with respect to the 16L8's logic diagram on the opposite page, where I've highlighted all the isolated 0s with a red 'X' for better clarity.

The pictorial form of the logic diagram is much easier to visualize than the text form of the JEDEC file but even then, determining which inputs are active and in-use is more tricky than determining the outputs by comparison. Add to this the error prone process of transferring from text to pictorial map which, in itself can also be a time-consuming task. This is by far the most direct way to identify used and unused pins of a PLD based on its JEDEC file, but there is a better alternative—one which allows you to skip the tedious step of doing text to pictorial map transfer—unless you're able to decipher the 1s and 0s directly from the JEDEC file, that is!

²⁰⁷ PLD design knowledge and experience are not necessary, but having them is certainly a bonus.

²⁰⁸ A JEDEC file (.jed) is a standard file format that specifies a fuse map that tells a PLD programmer how the array of fuses contained in a PLD is to be programmed.

²⁰⁹ This manual is readily made available online in the Data I/O website. Just search '3980xpi programmer user manual' and you should be able to find the link. The JEDEC kernel mode format is found on pages 5-37 and -38.

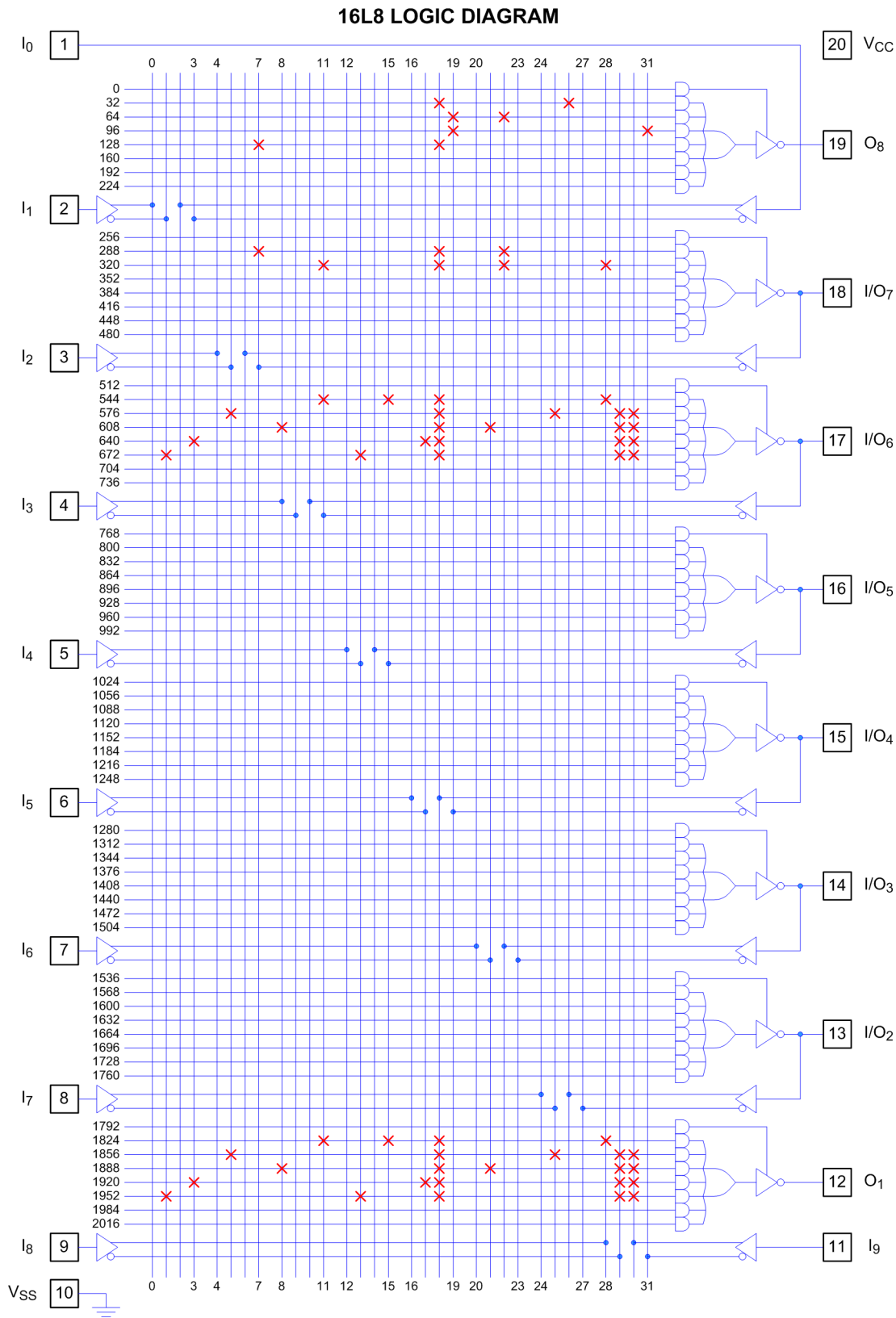
PAL MMI 16L8 EPP-80

```

*L00000 11111111111111111111111111111111
*L00032 11111111111111111011111110111111
*L00064 11111111111111111110110111111111
*L00096 111111111111111110111111111110
*L00128 111111101111111110111111111111
*L00160 000000000000000000000000000000
*L00192 000000000000000000000000000000
*L00224 000000000000000000000000000000
*L00256 111111111111111111111111111111
*L00288 11111110111111111110111011111111
*L00320 1111111111011111101110111110111
*L00352 000000000000000000000000000000
*L00384 000000000000000000000000000000
*L00416 000000000000000000000000000000
*L00448 000000000000000000000000000000
*L00480 000000000000000000000000000000
*L00512 111111111111111111111111111111
*L00544 1111111111011101101111111110111
*L00576 1111101111111111101111110111001
*L00608 11111111011111111101101111111001
*L00640 1110111111111111100111111111001
*L00672 10111111111101111011111111111001
*L00704 000000000000000000000000000000
*L00736 000000000000000000000000000000
*L00768 000000000000000000000000000000
*L00800 000000000000000000000000000000
*L00832 000000000000000000000000000000
*L00864 000000000000000000000000000000
*L00896 000000000000000000000000000000
*L00928 000000000000000000000000000000
*L00960 000000000000000000000000000000
*L00992 000000000000000000000000000000
*L01024 000000000000000000000000000000
*L01056 000000000000000000000000000000
*L01088 000000000000000000000000000000
*L01120 000000000000000000000000000000
*L01152 000000000000000000000000000000
*L01184 000000000000000000000000000000
*L01216 000000000000000000000000000000
*L01248 000000000000000000000000000000
*L01280 000000000000000000000000000000
*L01312 000000000000000000000000000000
*L01344 000000000000000000000000000000
*L01376 000000000000000000000000000000
*L01408 000000000000000000000000000000
*L01440 000000000000000000000000000000
*L01472 000000000000000000000000000000
*L01504 000000000000000000000000000000
*L01536 000000000000000000000000000000
*L01568 000000000000000000000000000000
*L01600 000000000000000000000000000000
*L01632 000000000000000000000000000000
*L01664 000000000000000000000000000000
*L01696 000000000000000000000000000000
*L01728 000000000000000000000000000000
*L01760 000000000000000000000000000000
*L01792 111111111111111111111111111111
*L01824 1111111111101110110111111110111
*L01856 1111101111111111101111110111001
*L01888 11111111101111111011011111111001
*L01920 11101111111111111001111111111001
*L01952 10111111111101111011111111111001
*L01984 000000000000000000000000000000
*L02016 000000000000000000000000000000
*C47FC

```

And you thought Thomas A. Anderson (aka Neo) was the only one who had to contend with the 1s and 0s in order to unravel the codes within the Matrix to become The One!



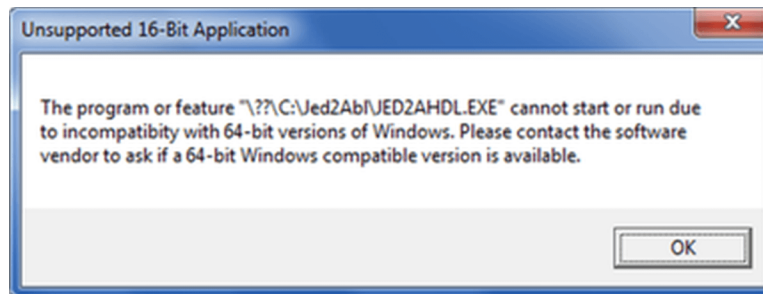
JEDEC De-compiler

The alternative method involves a DOS-based program called JED2AHDLE.EXE, which may require you to do a bit of setup and tinkering, depending on what version of Microsoft Windows you are running on your PC or laptop. We'll talk a bit about that a while later.

JED2AHDLE.EXE is basically a JEDEC de-compiler that can reverse-compile a JEDEC file into its device equations in ABEL's hardware description language (HDL) format,²¹⁰ supporting mostly simple PLDs of the P22V10 and P16V8 types but not CPLDs of any kind. First, you'll need to download the program and its support files from the following website: ²¹¹

<http://www.mikrocontroller.net/attachment/79696/Jed2Abl.zip>

Next, unzip it to C:\JED2ABL as default. If you're running Windows XP and earlier versions, or the 32-bit version of Windows 7, you should have no problem executing JED2AHDLE.EXE using a DOS command prompt window. However, running a DOS-based program inside 64-bit Windows 7 or higher versions does pose a separate set of challenges, unlike the earlier 16 and 32-bit versions of the operating system that still maintain a DOS layer for backward compatibility. You can still open a DOS command window in the 64-bit Windows 7 environment, but if you try to execute JED2AHDLE.EXE, it will prompt you:



To work around this limitation, there are two options:

- If you're running Windows 7 Professional, Enterprise, or Ultimate editions, you can install and use Windows XP Mode like a virtual operating system within the 64-bit Windows 7 environment to run DOS-based programs in the DOS command window,
- If you're running Windows 8 or the 64-bit Home editions of Windows 7, you can run a third-party DOS emulator or setup a virtual PC environment with a running copy of 32-bit Windows or earlier (such as Windows 2000 or 98), or even the basic MS-DOS operating system itself. In [Appendix E](#), I'll show you how to do both as well as state their pros and cons.

²¹⁰ It is beyond the scope of this book to explain the ABEL-HDL language, though the example provided later will show that it's not difficult to understand its syntax. However, for those readers who want to find out more, Lattice Semiconductor Corp. has made its [ABEL Design Manual](#) available online.

²¹¹ The link is from an attachment in a discussion forum dated 2010-06-10 and is still active at the time of writing. Should it become unavailable, try searching online and if all else fails, email me and I'll gladly send you a copy (it's a small 394kb zipped file).

Chapter 5

Once you're able to run JED2AHDL.EXE within the DOS command window, you'll be shown the syntax usage of this program:

```
C:\>jed2ahdl
```

```
JED2AHDL JEDEC to ABEL-HDL translator  
ABEL 6.00 Copyright 1983-1994 Data I/O Corp. All Rights Reserved.
```

```
Usage: JED2AHDL [-i] input_file[.jed] [options]
```

```
Options
```

```
-o output_file[.abl]  
-dev device_name  
-report map_file[.map]
```

As a minimum, you need a JEDEC file in the kernel mode (code 92) format as input for processing. You'll also need to specify the device name (see Appendix [E-1](#) and [E-2](#) for a complete list of supported PLDs). Using the fuse map JEDEC file listed earlier, under the filename U1.JED, which is a P16L8 type PLD, we can type the following at the DOS prompt:²¹²

```
C:\>jed2ahdl -i U1.JED -o U1.ABL -dev P16L8
```

```
JED2AHDL JEDEC to ABEL-HDL translator  
ABEL 6.00 Copyright 1983-1994 Data I/O Corp. All Rights Reserved.
```

```
Input JEDEC file is: u1.jed.
```

```
Output AHDL file is: u1.abl.
```

```
Device type is: p16l8.
```

```
Processing
```

```
  Reading Device Library
```

```
  Reading JEDEC Input File
```

```
  Extracting PLD Circuit Model
```

```
  Writing Output File
```

```
    DECLARATIONS Section
```

```
    EQUATIONS Section
```

```
    TEST_VECTORS Section
```

```
JED2AHDL complete. Time: 1 seconds.
```

An ABEL-HDL device equation file (.abl) will be created containing the de-compiled JEDEC content, which you can view with any text editor.

There are three sections within the ABEL-HDL file:

- **DECLARATIONS** Contains device, pin, node and ISTYPE as well as constant declarations.
- **EQUATIONS** Defines the functional and structural elements—equations, truth tables, state diagrams, fuses and XOR factors.
- **TEST_VECTORS** Used for simulation of the device's functional operation.

For our purpose, we're only interested in the EQUATIONS section (highlighted in [blue](#)).

²¹² Caps or no caps is irrelevant, but is shown here for clarity. If you choose to omit the output_file option (-o), a .abl file will still be generated with a similar filename as the JEDEC input file you specified.

Here is the ABEL device equation file de-compiled from the JEDEC file:

```

MODULE U1
"Created by JED2AHDL ABEL 6.00 on Wed Oct 15 13:45:21 19;4

TITLE
' PAL MMI 16L8  EPP-80 '

    U1 device 'p16l8';

"Pin and Node Declarations
    Pin01, Pin02, Pin03, Pin04 PIN  1, 2, 3, 4;
    Pin05, Pin06, Pin07, Pin08 PIN  5, 6, 7, 8;
    Pin09, Pin10, Pin11, Pin12 PIN  9,10,11,12;
    Pin13, Pin14, Pin15, Pin16 PIN 13,14,15,16;
    Pin17, Pin18, Pin19, Pin20 PIN 17,18,19,20;

    Pin12,Pin13,Pin14,Pin15,Pin16,Pin17,Pin18,Pin19 ISTYPE 'Com';
    Pin12,Pin13,Pin14,Pin15,Pin16,Pin17,Pin18,Pin19 ISTYPE 'Invert';

    X,K,Z,C,P,U,D = .X.,.K.,.Z.,.C.,.P.,.U.,.D.;

EQUATIONS

!Pin12 = (!Pin17 & !Pin16 & Pin15 & Pin09
    # !Pin03 & Pin15 & !Pin08 & !Pin09 & Pin11
    # !Pin04 & Pin15 & !Pin07 & !Pin09 & Pin11
    # !Pin01 & !Pin06 & Pin15 & !Pin09 & Pin11
    # !Pin02 & !Pin05 & Pin15 & !Pin09 & Pin11 );
Pin12.OE = ( 1 );

!Pin13 = ( 0 );
Pin13.OE = ( 0 );

!Pin14 = ( 0 );
Pin14.OE = ( 0 );

!Pin15 = ( 0 );
Pin15.OE = ( 0 );

!Pin16 = ( 0 );
Pin16.OE = ( 0 );

!Pin17 = (!Pin17 & !Pin16 & Pin15 & Pin09
    # !Pin03 & Pin15 & !Pin08 & !Pin09 & Pin11
    # !Pin04 & Pin15 & !Pin07 & !Pin09 & Pin11
    # !Pin01 & !Pin06 & Pin15 & !Pin09 & Pin11
    # !Pin02 & !Pin05 & Pin15 & !Pin09 & Pin11 );
Pin17.OE = ( 1 );

!Pin18 = (!Pin18 & Pin15 & Pin14
    # !Pin17 & Pin15 & Pin14 & Pin09 );
Pin18.OE = ( 1 );

!Pin19 = (Pin15 & Pin13
    # !Pin15 & Pin14
    # !Pin15 & !Pin11
    # !Pin18 & Pin15 );
Pin19.OE = ( 1 );

TEST_VECTORS
([ ]->[ ])

END

```

From the [EQUATIONS](#) section, we can clearly see that output pins 13, 14, 15 and 16 of U1 are not used, which is evident from the logic diagram we saw earlier, and also from the JEDEC listing rows [L00768](#) thru [L01760](#) containing all 0s. The remaining four output pins are used, of which pins 12 and 17 share similar equation, perhaps for the purpose of distributing the number of loads to drive instead of using just a single output which may not have sufficient sourcing strength. You can determine from the AND-OR Boolean expressions on the right hand side of the equations, the input pins that affect the output pins too. In the case of U1, all input pins are used.

So we see how JED2AHDL.EXE can really help us by converting a JEDEC file of the SPLD type into its device equations, so we can quickly ascertain used and unused input/output pins, freeing us from guess work to focus on the real task.

Disclaimer:

What we have covered in this section on deciphering the JEDEC code (or fuse map, if you prefer) applies only to PLD devices belonging to the simple PLDs (SPLDs) category, which are restricted to sum-of-product terms such as PALs and GALs. The more complex PLDs (CPLDs) and field-programmable gate arrays (FPGAs) such as those mentioned on page [37](#) are far too complicated and impossible to be deciphered in their native binary formats.

In most cases, due to their high pin-counts, the bulk of input/output signals are likely to be multiples of address or data bus lines and sequential in nature, compared to the combinatorial applications of the PALs and GALs.

Reverse-engineering a PCB containing CPLDs or FPGAs can be a pretty challenging undertaking and I would advise my readers not to attempt it without good reasons. This again begs the question as to how thorough should one go about reverse-engineering a PCB. This is not to discount the possibility of doing a complex PCB but whether it's worth the effort and time doing it. Having said that, what you've learnt from this book will certainly help you to better analyze each potential candidate and decide how best you want to do it—partially or completely.

SUMMARY

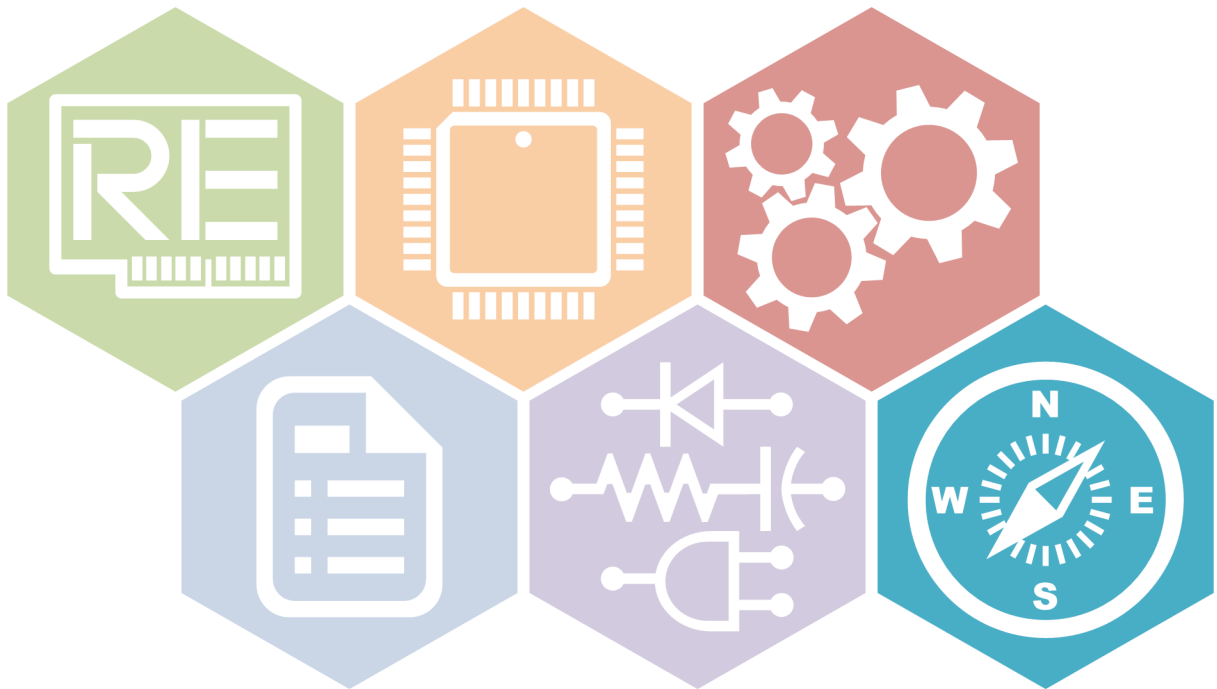
The above are some of the know-how which I've picked up through years of doing reverse-engineering, but I'm sure you'll be able to add on to the list as you gain expertise and experience in this exciting yet challenging journey to unravel the beauty of the original design. I'm glad you've made it this far with me and I hope you enjoyed reading and doing what has been put forth in my first book.

Before I sign off from here, I thought it'll be good to share some personal perspectives (and goodies) that will hopefully point you to some interesting directions for your next endeavor in the field of electronics...

6

Going From Here

It's definitely going to take more than luck to bring you to a higher level of success...



CONTENT

Fabulous Freebies–274

 KiCAD–274 DesignSpark PCB–276

Web-Based Wonders–277

 EasyEDA–277 Digi-Key EDA Design Tools –279

Old—but not Obsolete–280

 DOS-based OrCAD–280

Commercial Suites–281

 Altium Designer–282 Proteus Design Suite–284

Rantings on Circuit Simulation–286

Summary–287

6

GOING FROM HERE

"Have the courage to follow your heart and intuition. They somehow know what you truly want to become."

Steve Jobs, 1955-2011

There's a Chinese saying, "There is no limit to knowledge, and there is no satiety to study. So live and learn." I think Buzz Lightyear (in Pixar's animation film, Toy Story) puts it more succinctly in his famous line, "To infinity and beyond!" At the beginning of this book I did say that I'll not be teaching you to use EDA tools for reverse engineering. This does not, however, mean that I'm not for the idea that you learn to use them for that purpose. For some of you may already be using these tools in your daily work, and for others it could just be the next logical step to take—to reproduce the physical PCB after you have reverse-engineered it.

In fact, the first schematic diagram I created with was using a DOS-based EDA tool called SDT from OrCAD, back in the early 90's when the Intel 386 and the CRT monitor were the standard in desktop PC configuration. Most EDA tools such as those from Cadence and Mentor Graphics were still very much confined to expensive high-end workstations, and the license cost per seat is simply beyond the budget of non-PCB design engineers. OrCAD was the first to develop and sell low-cost, PC-based EDA software.²¹³ Of course, with the kind of processing power and graphics in today's PCs, there's definitely no shortage of EDA vendors entering the playing field now.

Different PCB designers will swear by the kind of EDA tools they are proficient in. Generally, commercial PC-based EDA software can be categorized into two types: hobbyist and professional. The former are usually low-end in terms of price and features, while the latter tend to be more expensive but comes with more powerful and integrated functions, as well as more extensive libraries and support. Due to stiff competition among these EDA vendors, it is not surprising that a number of them offer free, scaled-down or time-limited trial versions of their software to allow you to try-before-you-buy just to be sure it suits your requirement.

²¹³ OrCAD Systems Corporation started in Hillsboro, Oregon and became successful with their suite of DOS-based EDA tools like the SDT (Schematic Design Tool), VST (Verification and Simulation Tool) and PCB (which included a pretty decent autorouter!). Soon OrCAD broadened their range of products and migrated to Microsoft Windows, and in 1997-1998 merged with MicroSim Corp (flagship product PSPICE), before finally being acquired by Cadence Design Systems in 1999 when the merger didn't quite work out.

Still, there are some pretty good freebies out there, a couple of them from open source communities and a few from well-known electronic component distributors. And as of this writing, I have even come across web-based EDA tools that you can start using without having to download any package to install into your PC! Of course, there is naturally the concern about design copyright and confidentiality for using such free online software, so it is important to read and understand the terms and conditions laid down by the software providers before registering or signing up to use them.

In the pages that follow I will introduce some of these EDA tools, most of them free—downloadable or online—and one or two commercial-grade products worthy of mention.²¹⁴ The idea here is to broaden your horizon to the vast amount of available EDA resources available to you as an electronic engineer today, so you can make an informed decision on how you can further your knowledge and bring your skill set to a whole new level, while at the same time perhaps rekindle your enthusiasm in electronics once again if it had dwindled or died as a result of your mundane day-to-day work.

FABULOUS FREEBIES

The open source communities have been growing steadily and with it comes the benefit of free software developed by groups of volunteers and enthusiasts. Though many of these freebies are amateurish and experimental in nature and qualities, occasionally there's a gem or two that stands out with features that are comparable or even rival commercial products. One such product is KiCAD, an open source EDA software suite originally developed by Jean-Pierre Charras and currently under the care of the KiCAD developers team.

KiCAD²¹⁵

KiCAD runs in a number of popular OS, notably Microsoft Windows, Linux, and Apple OSX. This is pretty amazing considering a lot of PC-based commercial products only support the Windows OS, and for good reasons (\$\$\$). Size of download depends on your target OS as well, with the Windows version taking up slightly over 200MB for the stable [BZR4022](#) release (dated 2013-07-07).

Unlike commercial products, don't expect frequent updates or releases. That said, KiCAD has a strong following and the software is under continuous development and has reached a point of maturity that you can use it to do some serious PCB design work.

²¹⁴ Disclaimer: I don't represent any of these companies or products mentioned here, neither do I receive any cash, commission, or benefit whatsoever for introducing or even recommending them in my book. The reader is advised to exercise discretion and assume all risks and responsibilities for his or her decision in using the information that is provided as is. (Whew! That should take care of the boring legal stuff...)

²¹⁵ You can download KiCAD at <http://www.kicad-pcb.org/display/KICAD/Download> including the source code! For engineers into programming this is a very good way to learn the internals of EDA software for free. I remembered spending about \$350 to buy the SPICE 3E source code from the university of Berkeley back in 1992, because I was interested in analog circuit simulation—and I was just a junior engineer earning about \$1,200 a month back then!



1. **Schematic Capture.** KiCAD schematic editor Eeschema lets you create complex hierarchical or flat schematic diagrams that span multiple sheets and levels using its comprehensive open-source text-based component library and in-built electrical rules check (ERC) utility.
2. **Component Association.** This is accomplished using Pcbnew to associate each single schematic symbol with its desired or equivalent component footprint before PCB layout.
3. **PCB Layout.** The Pcbnew board editor easily handles up to 32 copper layers and 12 engineering layers (silk screen, solder mask, etc.) and its auto-placement and built-in auto-router makes it a snap to layout simple to moderate PCB designs. More complex designs can be done by hand with its push and shove and track highlighting features.

The additional schematic and layout library component editors allow you to create or modify schematic symbols and component footprints, while the 3D viewer allows you to render a final 3D model of your PCB. Pcbnew and Gerbview are used to generate the production files (Gerber, NC drill and component location) for manufacturing your printed circuit board. Postscript or PDF file generation is also possible.

What's lacking in KiCAD right now is a circuit simulator tool. For an EDA suite that rivals commercial products in terms of power and features, it's kind of a letdown—but I suppose you can't ask too much for something this good and free. There is a not too elegant workaround though, and that is to use its Eeschema tool to generate a SPICE compatible netlist and then run the simulation off a SPICE-based simulation software.²¹⁶

Unless you're a design engineer, the need for a circuit simulator should be farthest from your mind at this moment, so KiCAD with its fundamental set of tools should be more than adequate to meet your requirement for now.

DesignSpark PCB

In its own words, DesignSpark PCB is the world's most accessible electronics design software. Easy to learn and easy to use, DesignSpark PCB is designed to significantly reduce your concept-to-production time. At the core of this unique approach is a powerful software engine that enables you to capture schematics, design PCB boards and layouts.

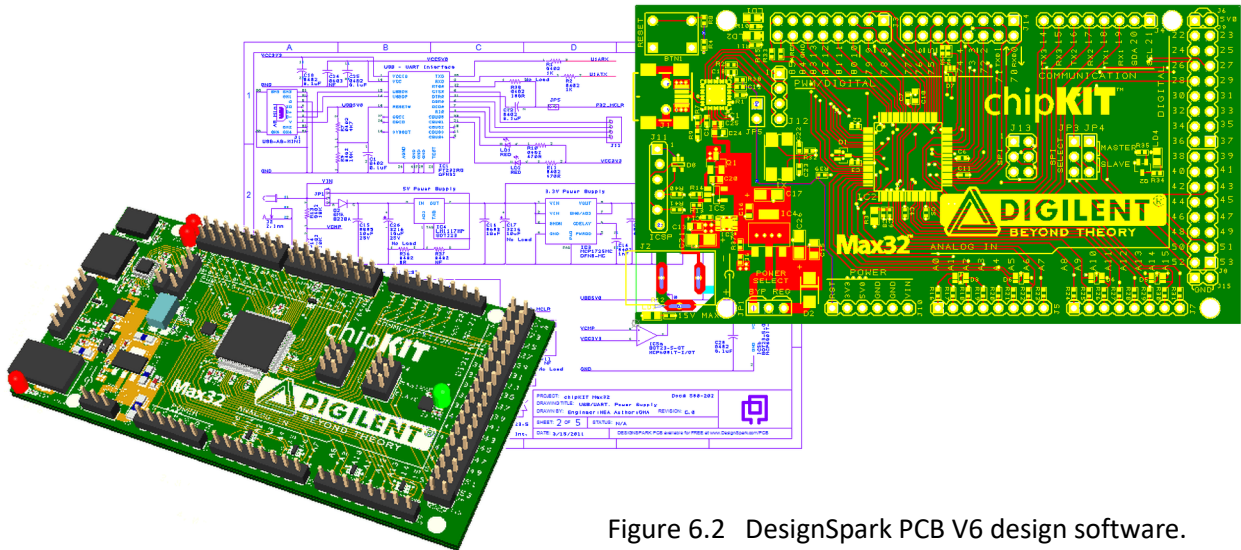


Figure 6.2 DesignSpark PCB V6 design software.

²¹⁶ There are a number of free SPICE simulators out there, but most of them only handle analog circuits. To be sure, there are free digital circuit simulators as well but you'd be hard pressed to make the two work together if you want to simulate a circuit that has analog and digital elements. Now, some of you may be thinking of using EDIF netlist as a workaround, but I need to warn you upfront that it's going to be a tough undertaking just to manage the data preprocessing part, let alone trying to ensure syntax compatibility as well as model parts library availability for the two separate simulators. The solution—a mixed-signal (aka mixed-mode) circuit simulator. Thankfully there are a few freebies out there and I will mention one of them later.

What's interesting about DesignSpark PCB, though free and unlimited for use, is that it is owned by RS Components and Allied Electronics, one of the world's largest distributors of electronic and maintenance products. After you've downloaded and installed the software, you'll need to register as a member in order to activate it.

Like KiCAD, the RS/Allied's DesignSpark Team is committed to the continuous development of their free PCB design software, taking feedbacks from worldwide users seriously to implement requested features and improvements, and have reached the current version 6 since its launched in mid 2010. Schematic capture, component creation, design rule check, import from Eagle (PCB, schematics, libraries), export to simulation tools (LTSpice, Tina, etc.), auto-placement and auto-router, generation of PCB production files (Gerber, Excellon, DXF, IDF, automated assembly data), you name it—DesignSpark PCB has it.

The best part of the deal is the software libraries are aligned with RS Components' product offer which is estimated around 80,000 components, which means you can generate a BOM from your PCB design furnished with RS order code, get a direct quote and place order for the parts without skipping a beat. Heck, they even do prototype quantities through their PCB Quote service. Talk about having your design needs covered from start to finish—no wonder they provide their software free of charge!

WEB-BASED WONDERS²¹⁷

If you think it's a hassle to download those big setup files and having to install into your PC just to try out the features, look no further because there are free online EDA tools as well. Ah, the wonder of high-speed internet age!

EasyEDA

As the name implies, it's an easy to use web-based EDA tool—schematic capture, circuit simulation, PCB layout—all in one. No installation required, just a web browser that's HTML5 capable and you're ready to go! EasyEDA boasts that it has all the features you expect and need to rapidly and easily take your design from conception through to production, and for good reasons too:

- Schematic Capture. Multiple-sheet flat and hierarchical schematics (passive drawings and active simulation schematics), as well as simple but powerful general drawing capabilities. Work can be saved in pdf, image or SVG format.
- Circuit Simulation. NgSpice-based simulator that supports SPICE models and sub-circuits, and accepts SPICE-compatible netlist. Waveform viewer that displays post simulation measurements and saves simulation plot data in CSV format.
- PCB Layout. Accepts PCB netlist formats from popular commercial EDA packages like Altium Designer and PADS, and even the free KiCAD! PCB design rules and checking included.

²¹⁷ The fact that they're online software means that you can work in any kind of OS (Linux, Mac or Windows) from within any modern web browser (Chrome, Firefox, IE, Opera, or Safari).

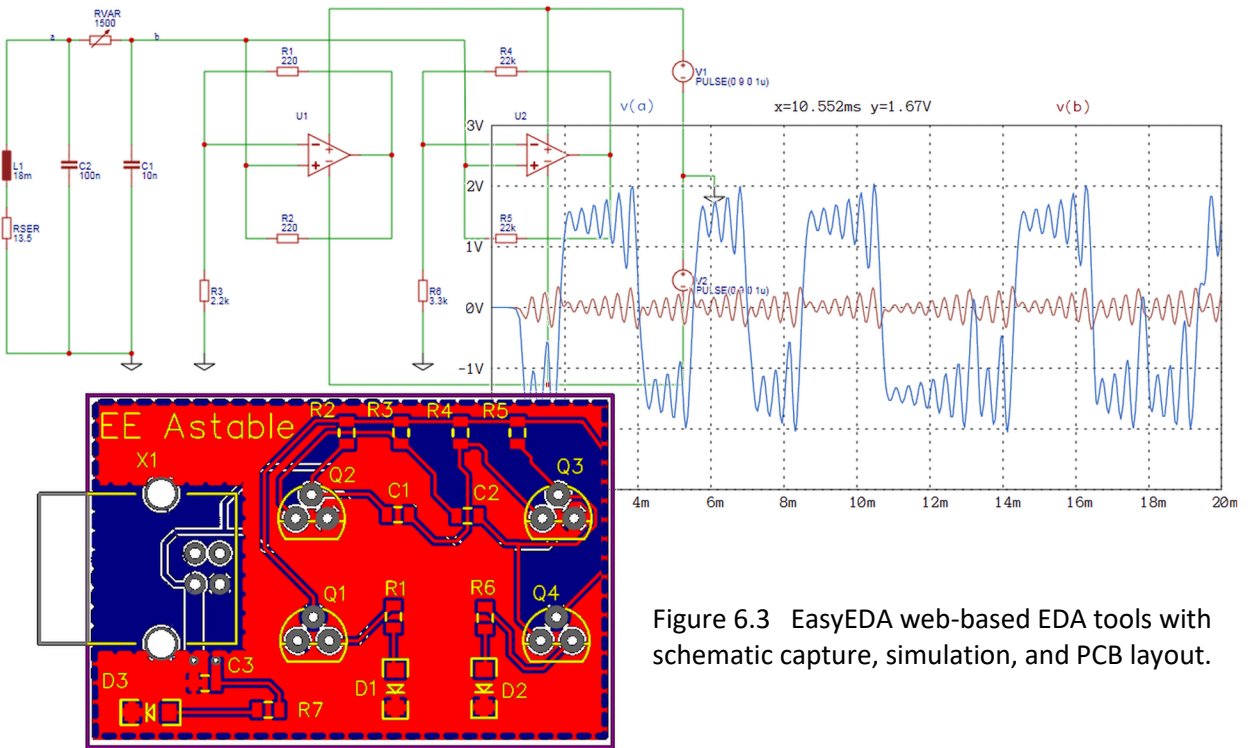


Figure 6.3 EasyEDA web-based EDA tools with schematic capture, simulation, and PCB layout.

- Library Management. Schematic symbols, SPICE sub-circuits, and PCB footprints creation and editing, or import from other EDA software libraries (Altium, Eagle, LTSpice, KiCAD).
- Creation of BOM reports.
- Online sharing of—and collaborative working on—schematics, simulations, PCB layouts, designs and projects.

While you can still use most of the features of EasyEDA without signing up for an account, you're limited to saving files in Anonymous mode and can only re-open and edit your work by pasting the urls into the browser. After you've signed up and opened an EasyEDA account, the more powerful file management and sharing abilities become accessible to you.

Unlike some of the free PCB design tools that tie you to their PCB production house, you can create and download your Gerber, NC drill and component files from EasyEDA and send them anywhere you like. A major feature of the EasyEDA revenue model is that it provides a low cost PCB fabrication service. Just for comparison: Express PCB²¹⁸ charges \$98 for three 4-layer boards (3.75" x 2.5") + shipping, while EasyEDA charges \$25 for five 4-by-4 inch double-layer boards.

As with any online software, you need to consider the confidentiality aspect when using the tools to do your work. Common sense is the best guide.

²¹⁸ One of the free PCB layout software vendors that also provide PCB fabrication services.

Digi-Key EDA Design Tools



Figure 6.4 Digi-Key's EDA design tools.

Similar to RS Components, Digi-Key in collaboration with Aspen Labs, also offers a free design software suite that includes the standard tools like schematic capture, circuit simulation, and PCB layout, except that it's online instead of a download and install package. Digi-Key claims that its software enables true back-of-the-napkin to bill of materials (BOM to BOM) functionality, through analog/power simulation, all the way to the most robust professional full PCB schematic capture and layout functionality, bringing PCB designs to fruition quicker and easier than ever before. And as expected, the software is linked to Digi-Key's inventory of electronic components so that after you're done with your design, hopefully you'll buy the parts from them as well.

A quick run through will suffice:

- Scheme-it is an online schematic drafting and block diagramming tool for designing and sharing electronic circuit diagrams, and includes a comprehensive electronic symbol library integrated to Digi-Key's component catalog that allows for a wide range of circuit designs. Additionally, a built-in bill of materials manager is provided to keep track of parts used in a design. Once a schematic drawing is complete, users can export it to an image file or share it via email with others. Scheme-it works natively in all major web browsers without requiring the use of any plug-ins. Only registered users are able to share and save designs.
- PartSim is an easy to use SPICE circuit simulator that runs in a web browser, supporting AC, DC, and transient simulations, and comes with a waveform viewer.
- PCBWeb is a PCB layout tool that can route multi-layer boards with support for copper pours and DRC checking. Ability to order PCBs.

I've not personally tried this EDA tool but I'm quite certain that it should be similar to DesignSpark and EasyEDA to some degree.

OLD—BUT NOT OBSOLETE

Good things never die, even if they grow old. The same can be said about software, one in particular which I've fondly remembered since my early engineering days...

DOS-Based OrCAD

You might not believe me if I tell you that there's a community of PCB designers who're die-hard DOS-based OrCAD fans.²¹⁹ For readers who've worked with the DOS version of OrCAD products, it may come as a pleasant surprise that these EDA tools, which used to cost from \$495-\$1,995 individually or \$3,995 for a complete suite, are available for download complete with user and reference manuals. You'll need to register as a member first—subject to approval by the moderators of the group, though.

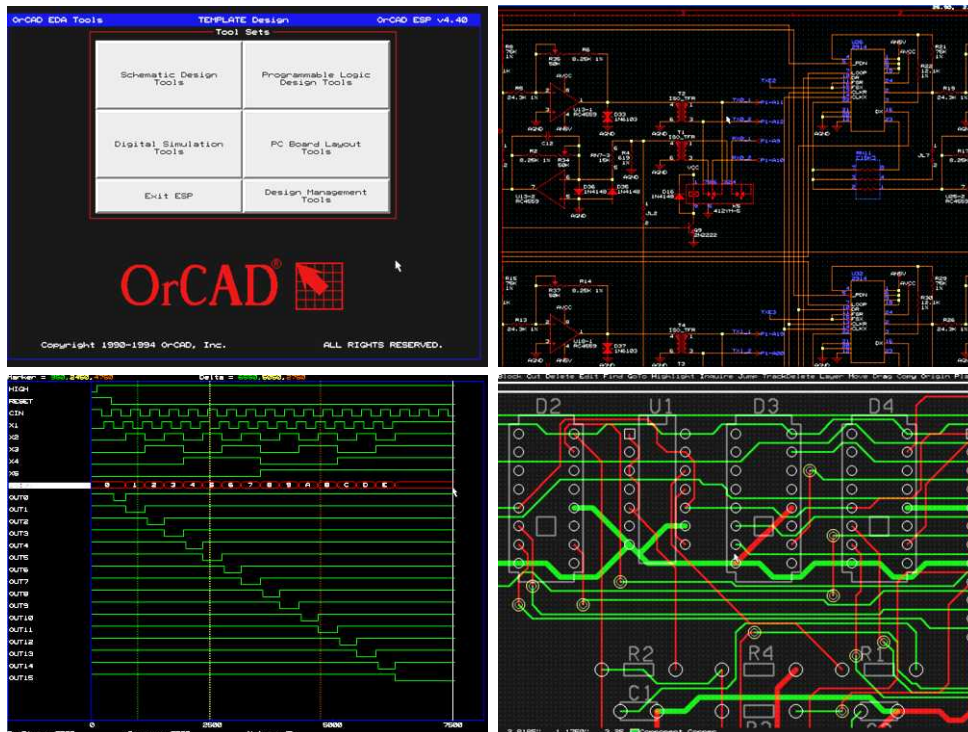


Figure 6.5 OrCAD 386+: Main menu, SDT, VST, and PCB.

The last DOS version is the OrCAD ESP 4.40 release which comprises the following modules:

- SDT 386+ 1.21 Schematic Design Tool
- VST 386+ 1.20 Verification & Simulation Tool
- PCB 386+ 2.22 PCB Layout Tool
- PLD 386+ 2.01 Programmable Logic Design Tool
- MOD 386+ 4.04 PLD Modeling Tool

²¹⁹ Just Google [Old DOS OrCAD](#) and you'll find the Yahoo! Group where the community is found.

I've worked extensively with the SDT 3.21 in my early engineering days from 1990-1995, mainly to draft schematics as well as flowcharts and 2D mechanical drawings for my test program set documents. To the hardware engineers in my company's engineering design department, this humble piece of software was just a toy compared to their powerhouse EDA suite from Mentor Graphics running on Sun's top-end graphical workstations. Perhaps, but then again there are many products out in the market which were designed and produced using OrCAD too.

So how do we run these legacy DOS software from within modern operating systems that are now the dominant landscape of most PCs today? The answer: Virtual PC or DOS emulator. And there are many flavors to choose from—Microsoft's Virtual PC, Oracle VM's VirtualBox, VMware Player, the open source DOSBox, etc. Most of these x86 machine emulators are free and they support and run different OSes such as Linux, Windows, OS/2, BeOS, and even the Mac OS X.²²⁰ I've installed and run Microsoft's Virtual PC and DOSBox with some of my favorite software of by-gone era just to re-live the good old nostalgic DOS days. You can find information on how to obtain and configure them in [Appendix E3](#) through [E13](#) should you want to give them a try.

COMMERCIAL SUITES

And now for the commercials... With the proliferations of free EDA software, many EDA vendors are facing stiff competitions to come out with better tools and support to get a piece of the market share. It is inevitable that those who do not re-invent their products to meet the increasing demand of a new generation of PCB designers and engineers will find themselves out of the race quickly.

In today's EDA software landscape, established names like Agilent, Cadence Design Systems, Cadsoft, Mentor Graphics, Synopsys and Zuken seem to be holding out well, while others either folded, merged, or were acquired by their rival companies. Familiar product names like OrCAD/Allegro, Protel, PADS, Cadstar, Eagles, MultiSim and Ultiboard still remain favorites among PCB designers and engineers, and I had the opportunity to try my hands on some of them too.

However, there are two products that stand out strongly in recent years and gaining large followings—and for very good reasons:

- Firstly, their flagship products underwent major overhauls to include powerful tool sets and advanced features to take advantage of modern hardware and OS capabilities;
- Secondly, their parts libraries are pretty comprehensive, with integrated support for many EDA vendors' libraries and work files format import and export;
- Thirdly, their products support firmware and embedded development, either through real-time simulation of processors and microcontrollers or physical hardware development platforms that interact with the software simulator; and
- Lastly, they provide comprehensive documentation and/or tutorial videos online freely, besides downloadable time-limited full demo versions, so you can learn key concepts and try out their software and even ask legitimate real-world design questions that they'll gladly answer!

²²⁰ These are exciting times we live in where the possibilities of running multiple OSes within a single platform (PC or MAC) is no longer a dream, putting the power of unparalleled computing and learning in the hands of engineers and individuals, which was unthinkable about a decade ago.

Now, I won't go into those boring details of the standard packages; instead, I'll highlight the features that make these two products stand out in the crowd. Let's take a look at them now...

Altium Designer

The ability to view your design project from different perspectives all at the same time, while seamlessly navigating and editing with real-time synchronization, is a PCB designer's dream—and that's just what Altium Designer excels in with its native multiple monitors work environment since the original Protel DXP release in 2004.

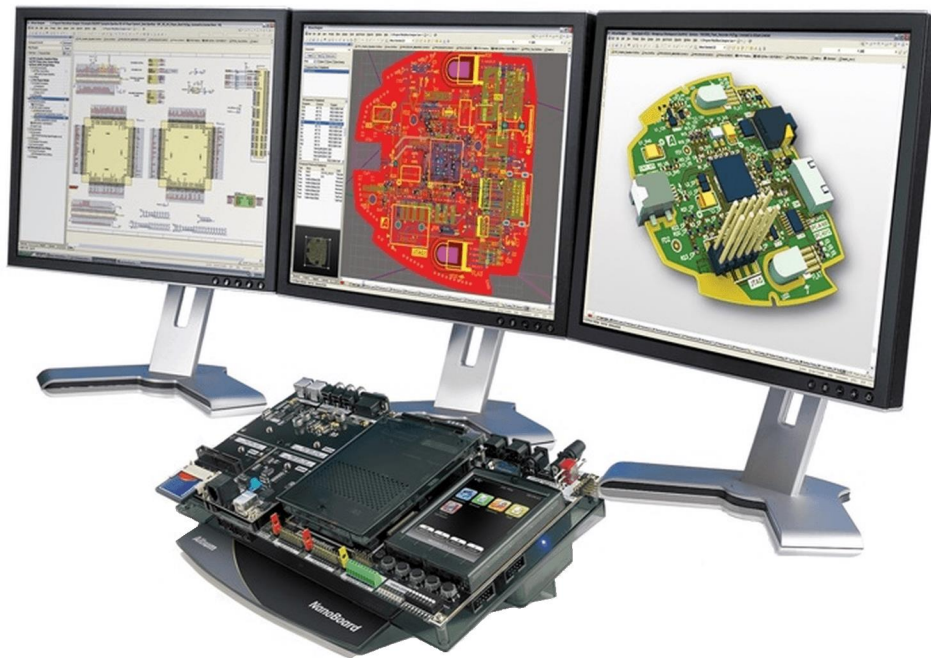


Figure 6.6 Altium Designer and NanoBoard hardware platform.

But Altium Designer does more than just dazzle with a pretty user interface; its unified approach in board design, smart project data management, firmware design and rapid prototyping, and extension support for many hardware vendors makes it easy for third party developers to integrate their own product functionalities into Altium's already impressive EDA tools arsenal. And that's not all. The latest release added capabilities for flex and rigid PCB design, enhanced layer stack management, embedded components within PCB layer stack, and true variant support which is not limited to just component parameters change but component replacement as well.

Altium Designer is not limited to just software for its design products. Altium's NanoBoard range of hardware development platforms incorporated field-programmable gate array (FPGA) technologies for implementing and debugging FPGA designs as well, providing communications to and from the on-board FPGA that interacts with a wide range of peripherals including LCD, flash memory, RAM,

keyboard, etc. for rapid application development.²²¹

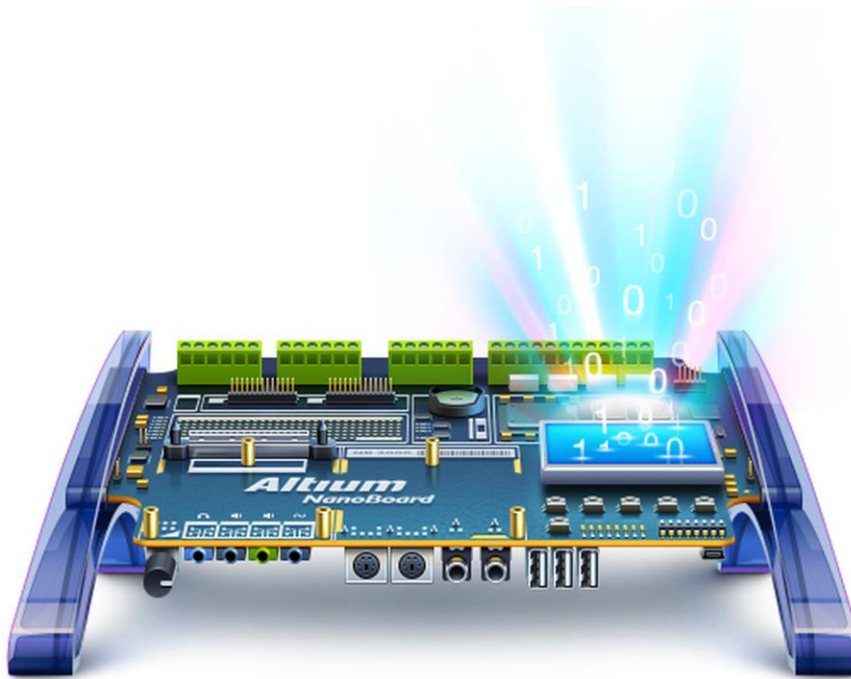


Figure 6.7 One of Altium's 3000-series NanoBoards.

Though I do not personally own or use Altium Designer,²²² I was fortunate to attend a free seminar organized by a local authorized dealer and training centre last year, who flew an Altium staff engineer all the way from the US to conduct a one-day product demonstration for their 2013 release. It was an eye-opening and educational experience to witness firsthand the software's impressive performance. Each participant was also given a DVD package containing a trial copy and complementary tutorial videos; one of them even walked away with an NB2 model NanoBoard from the lucky dip!

So if your company is into the PCB design business or thinking about going into it, you may want to take a look at what Altium has to offer.

²²¹ These NanoBoards can also be chained together for complex system development. As of this writing, Altium has introduced its 3000 series of NanoBoards which spot FPGAs from well-known vendors like Xilinx, Altera and Lattice.

²²² The price for a license starts at \$7,245 when Altium Designer 14 debut in late 2013 but it might have changed since. It's still reasonably affordable compared to Cadence's PCB Designer Professional which, according to some of its users, can cost even more for a useful configuration that allows you to do complex PCB designs.

Proteus Design Suite

The Proteus Design Suite from Labcenter Electronics offers the ability to co-simulate both high and low-level microcontroller code within a mixed-mode SPICE circuit simulator via its Virtual System Modeling (VSM), a collection of animated components and a rich array of virtual instruments. Its extensive single step and debugging facilities include system wide diagnostics, support for a wide selection of processors and microcontrollers,²²³ and the ability to work with popular compilers and assemblers, allowing your code to interact with simulated hardware almost real-time. The result is a streamlined product design cycle, reduced time to market and lower costs of development.

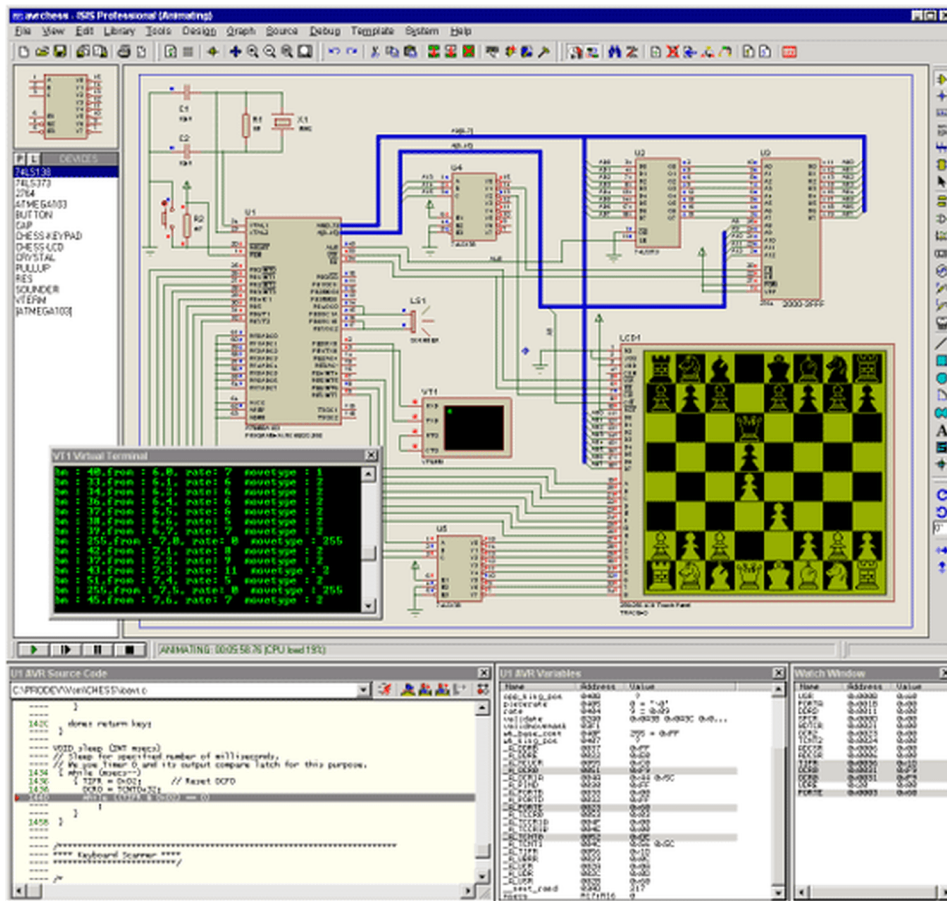


Figure 6.8 Proteus Design Suite with co-simulation capabilities.

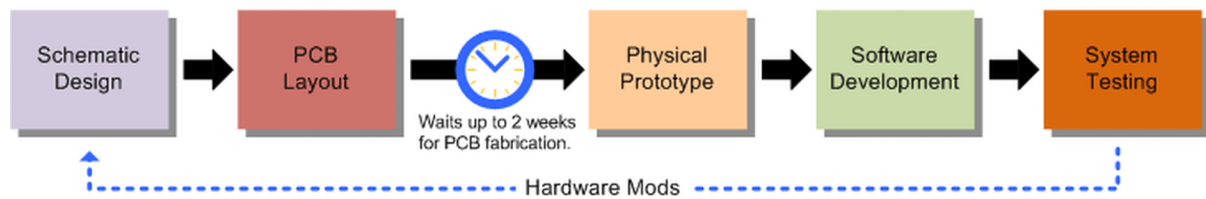
Figure above shows a VSM simulation of a 'Tiny Chess' design using an Atmel AVR MEGA103 processor together with graphical LCD, keypad, sounder/buzzer and virtual terminal models. It also shows the design being debugged with the source, variables and watch window along the bottom of the screen. So with VSM, it is possible to develop and test such designs before a physical prototype is constructed.

²²³ Available for PIC, 8051, AVR, HC11, ARM7/LPC2000 and Basic Stamp processors.

I've chosen to highlight Proteus' VSM features instead of its schematic capture (ISIS), PCB layout (ARES), library editors and advanced SPICE-based simulator (ProSPICE),²²⁴ because the VSM is what sets it apart from the rest of the competitors, including the well-known Electronics Workbench EDA software from National Instruments. VSM goes beyond other circuit simulators that provide animated components and virtual instruments; it supports a broad range of microcontroller and processor models²²⁵—and the vast majority of compilers on the market—to facilitate near real-time simulation of microcontroller-based designs—even those that contain multiple CPUs!

The VSM Advantage

Traditional Design Tools



Proteus VSM

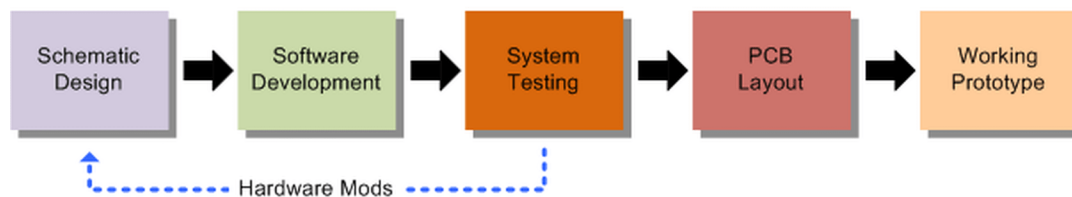


Figure 6.9 The difference between traditional and VSM approaches.

With traditional design tools, software development and system testing cannot begin until a PCB design and physical prototype are available, incurring a delay of up to 2-3 weeks. If the hardware design does not work, the whole process will have to be repeated. With VSM, software development can begin as soon as the schematic is drawn—the combination of hardware and software can be thoroughly tested before physical prototyping.

If complex circuit simulation is what you're looking for besides PCB design, you may want to consider Proteus Design Suite. For more information, go to Labcenter Electronics website where you'll find time-limited trial copy to download and try, descriptive write-ups of the various tools and features, and even demo and tutorial videos to watch.

²²⁴ The Advanced Simulation Option provides a full range of graph-based analyses like analogue, digital and mixed-mode transient, frequency, AC and DC parameter sweeping, Fourier, noise, distortion, transfer curve, and digital conformance analyses, plus Labcenter's BASIC-like EasyHDL scripting language.

²²⁵ Supported hardware includes PIC family (10, 12, 16, 18, 24) and dsPIC 33, ARM LPC2000, Cortex-M3, MSP430, Piccolo DSP, Atmel AVR, Arduino AVR, 8051/52, HC11 and BASIC Stamp processors.

RANTINGS ON CIRCUIT SIMULATION

Finding a good circuit simulator that suits your style—yet powerful and easy enough to use—does take a bit of an effort. Proponents of SPICE-based simulators will insist that accurate simulation using real models of non-linear components and fine tuning their associated parameters is the way to go, though this approach usually has steep learning curves that discourage or even frighten away many would-be circuit designers.²²⁶ The good news is today's circuit simulators, even the free ones, recognizes this fact and try to minimize the pre- and post-processing steps so the user can get straight to simulating and analyzing their design without having to struggle with those unnecessarily complex steps to master the simulator software.

As an electronic engineer, you'd probably have designed some simple circuits, either by breadboarding or prototyping, and perhaps manually routed and etched your own double-layered PCBs, or have wire-wrapped using vero-boards. If you've not played with circuit simulators before, I assume you'd built your circuits from application notes found in datasheets or textbooks, or did your calculations by hand before implementing in hardware. Well, I've done my fair shares of all the above until my first encounter with Microcap II.²²⁷ That's when I learned firsthand what virtual breadboarding and circuit simulation is all about. Though it's schematic capture is primitive by today's standard, it's simulation options was by no means dated. Below is an AC analysis of a simulation run for a simple RC network using a sinusoidal source:

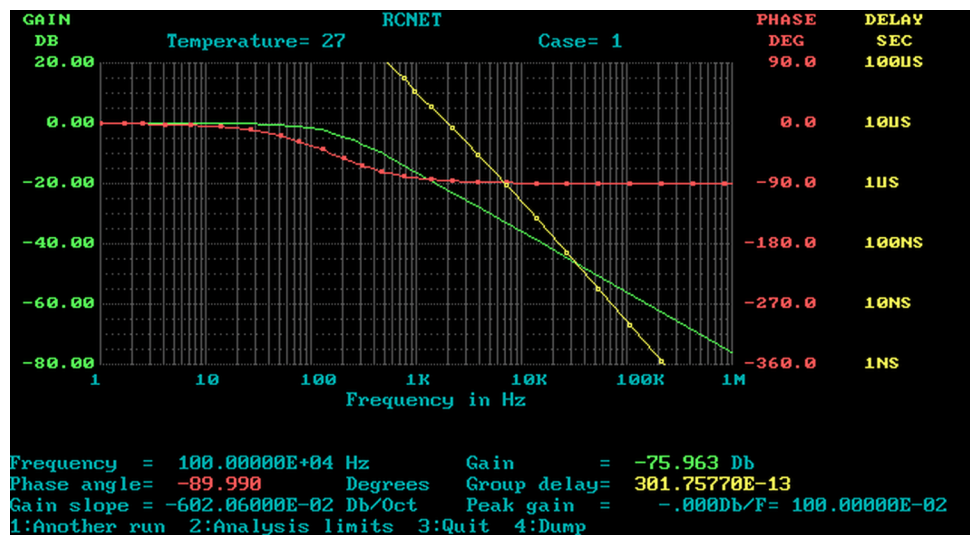


Figure 6.10 Microcap II AC analysis run of a simple RC network.

²²⁶ I remember attending a 3-day training course many years ago on Daisy/Cadnetix's early version of its DAZIX simulation tool on Sun Microsystems' workstation. It was interesting, challenging and often frustrating trying to work through the intricacies of modeling those analog parts and grappling with the right simulation parameters.

²²⁷ It's a student version that limits the node count and speed of algorithmic computations. Microcap II had a twin brother MicroLogic II but I wasn't as impressed because back then I was doing digital simulations on a Sun Sparc II 360 server running the HHB Systems CADAT simulator with concurrent fault simulation capability as well as a CATS 10000 hardware modeler. Try to beat that!

Allow me to indulge a little into nostalgia—don't you just love its simple yet elegant display presentation of the simulation results? You could even see the numbers running as the graph is being plotted—something that other SPICE programs back then could not do—which it did well and effortlessly within the confines of the DOS environment.²²⁸ This, and other experiences that I have shared throughout the book, are what motivated me to go a step further each time I find myself seemingly stagnant and not making fresh progress in my quest and knowledge in electronics.

I've already mentioned a couple of circuit simulators in the preceding sections, and I hope by now you have tried one or two to get a feel of what circuit simulation is all about, and how it can assist you in your circuit design work.

SUMMARY

The aforementioned EDA suites—free or commercial—each has its own combination of tool sets. While some try to be jack-of-all-trades by covering every aspect of the EDA design process from schematic capture, parts modeling, to circuit analysis and simulation, onto PCB layout and routing, as well as generating the files for PCB production, others simply choose to limit themselves to what they are good at by focusing on the essentials. Today's engineers are indeed spoilt for choices!

It is my hope that this chapter, with its brief write-ups of these different EDA software, will inspire you to explore and take up learning some of these tools, and in the process enrich your knowledge and keep the fire of love for electronics burning bright—a life-long pursuit in itself, both as an engaging hobby and successful career.

All the best to you, my reader!

²²⁸ Microcap is now into version 11 and is definitely better than ever— except for its price of \$4,495 which is really not affordable to hobbyists or the average engineer. A free limited demo version is available for download. Go to Spectrum Software's website for more information.

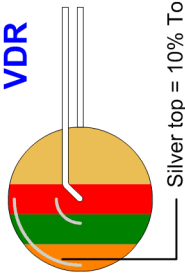
Appendix A

ELECTRONICS REFERENCES

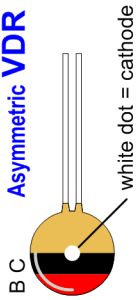
COLOR CODE FOR RESISTORS AND CAPACITORS	A-1
SMD SIZES FOR RESISTORS AND CAPACITORS	A-2
RESISTOR NETWORK CONFIGURATIONS	A-3
SMD CODES MARKING REFERENCE (Sample)	A-4
EIA MARKING CODE FOR SMD RESISTORS	A-5
CHIP CAPACITOR MARKING CODE TABLE	A-6
STANDARD MICROCIRUIT CROSS REFERENCES	A-7
SEMICONDUCTOR MANUFACTURER LOGOS	A-10

COLOR CODE FOR RESISTORS AND CAPACITORS

VDR



Asymmetric VDR



Miniature Ceramic Plate Tuning Capacitors (TC)



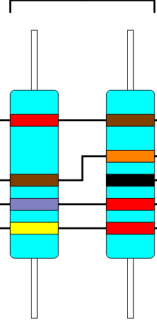
+100 X 10 ⁻⁶
0 X 10 ⁻⁶
-75 X 10 ⁻⁶
-150 X 10 ⁻⁶
-220 X 10 ⁻⁶
-330 X 10 ⁻⁶
-470 X 10 ⁻⁶
-750 X 10 ⁻⁶
-1500 X 10 ⁻⁶

±2% Tol.

Carbon Film Resistors



Metal Film Resistors



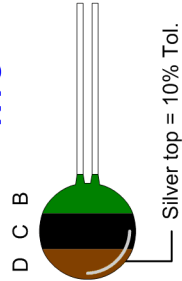
A	B	C	D	TOL
0	0	0	x1	±1%
1	1	1	x10	±2%
2	2	2	x100	±5%
3	3	3	x1K	±10%
4	4	4	x10K	±1%
5	5	5	x100K	±2%
6	6	6	x1M	±5%
7	7	7	x0.1	±10%
8	8	8	x0.01	±20%
9	9	9	x0.01	±0.1pF
			Ω	±0.25pF
			pF	±0.5pF
				±1pF

Standard Capacitors

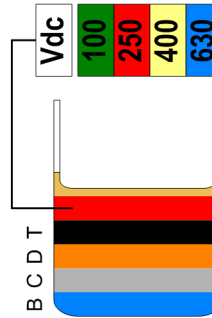


For resistors, absence of tolerance band indicates a 20% tolerance.
For capacitors, refer to data for the specific typed.

NTC

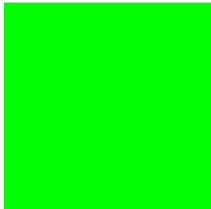


Flat Film Capacitors



Drawn by: NG KENG TIONG

SMD SIZES FOR RESISTORS AND CAPACITORS

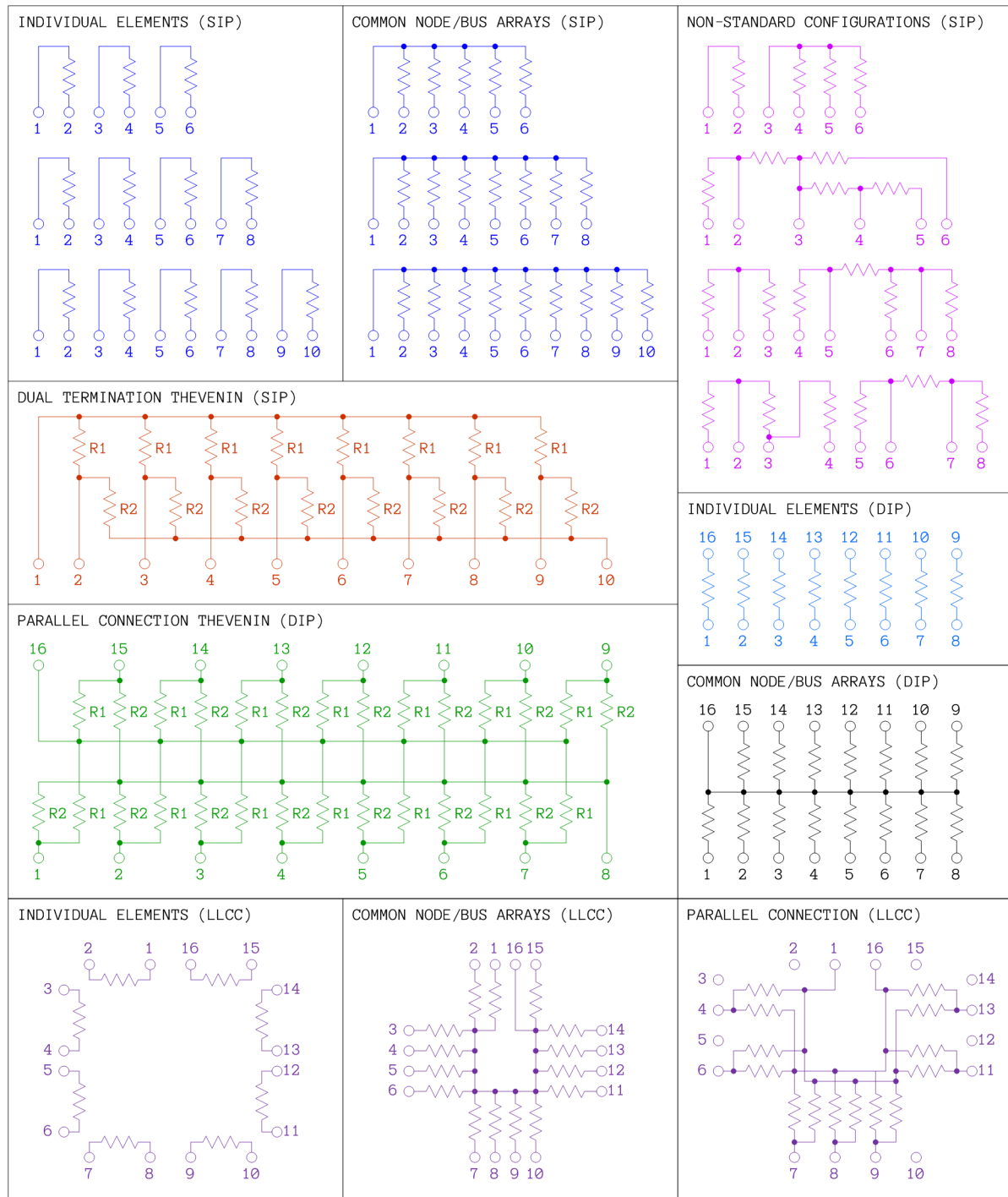
	Metric code	Actual size	Imperial code	
 0.1x0.1mm	0402	-	01005	 0.01x0.01 in (10x10 mils)
	0603	-	0201	
	1005	-	0402	
	1608	-	0603	
 1x1mm	2012	-	0805	 0.1x0.1 in (100x100 mils)
	2520	-	1008	
	3216	-	1206	
	3225	-	1210	
 1x1cm	4516	-	1806	 0.5x0.5in (500x500 mils)
	4532	-	1812	
	5025	-	2010	
	6332	-	2512	

Sizes shown are magnified approximately 2 times for better overall clarity. Metric and imperial codes denote dimensions in length and breadth according to their respective units:

Examples: Metric code 1608 = 1.6mm x 0.8mm
 Imperial code 0603 = 0.6inch x 0.3inch

(Source: Wikipedia)

RESISTOR NETWORK CONFIGURATIONS



(Drawn by Ng Keng Tiong)

SMD CODES MARKING REFERENCE (Sample)

Codes beginning with 'A'

Code	Device	Mftr	Base	Package	Leaded Equivalent/Data
A	BA892	Sie	I	SCD80	35V 100mA pin
A	1SS355	Roh	I	USM	100V 50mA sw
A	MRF947	Mot	N	SOT323	nnp RF 8 GHz
A0	HSMS-2800	HP	C	SOT23	HP2800 schottky
A0	HSMS-280B	HP	C	SOT323	HP2800 schottky
A03	VAM-03	MC	AQ	-	modamp MAR 3 Similar
A06	VAM-06	MC	AQ	-	modamp MAR 6 Similar
A1	BAW56W	Phi	A	SOT323	dual ca BAW62 (1N4148)
A1	BAW56	Phi	A	SOT23	dual ca BAW62 (1N4148)
A1	BAW56	Phi	A	SOT23	dual ca BAW62 (1N4148)
A1p	BAW56	Phi	A	SOT23	dual ca BAW62 (1N4148)
A1t	BAW56T	Phi	A	SOT416	dual ca BAW62 (1N4148)
A1t	BAW56S	Phi	-	SOT363	dual ca BAW62 (1N4148)
A1s	BAW56W	Sie	A	SOT323	dual ca BAW62 (1N4148)
A1s	BAW56	Sie	A	SOT23	dual ca BAW62 (1N4148)
A1s	BAW56U	Sie	A	SC74	dual ca BAW62 (1N4148)
A2	HSMS-2802	HP	D	SOT23	dual HP2800
A2	HSMS-280C	HP	D	SOT323	dual HP2800
A2	BAT18	Phi	C	SOT23	BA482
A2s	BAT18	Sie	C	SOT23	BA482
A2	MMBD2836	Mot	A	SOT23	dual ca sw diode 75V
A2	CFY30	Sie	CQ	SOT143	n-ch GaAsfet 6 GHz
A2	MBT3906DW1	Mot	DO	SOT363	dual 2N3906
A22	BAS21	Phi	C	SOD27	BAV21
A2X	MMBD2836	Mot	A	SOT23	dual ca sw 75V 100mA 15ns
A3	BAP64-03	Phi	I	SOD323	3 GHz pin diode
A3	HSMS-2803	HP	D	SOT23	HP2800 ser pair
A3	BAS16	Zet	C	-	Si sw 75V 100mA
A3	BAT17	Phi	C	SOT23	BA481
A3p	BAT17	Phi	C	SOT23	BA481
A3t	BAT17	Phi	C	SOT23	BA481
A3	MBT3906DW	Mot	N	SOT363	dual 2N3906
A3X	MMBD2835	Mot	A	SOT23	dual ca sw 35V 100mA 15nS
A4	HSMS-2804	HP	B	SOT23	dual cc HP2800 schottky
A4s	BAV70W	Sie	B	SOT323	dual cc BAW62
A4s	BAV70	Sie	B	SOT23	dual cc BAW62
A4s	BAV70T	Sie	B	SOT416	dual cc BAW62
A4p	BAV70	Phi	B	SOT23	dual cc BAW62
A4t	BAV70	Phi	B	SOT23	dual cc BAW62

(Source: GM4PMK's SMD Codebook)

EIA MARKING CODE FOR SMD RESISTORS

Digits	S/Y	R/X	A	H/B	C	D	E	Digits	S/Y	R/X	A	H/B	C	D	E
01	1R0	10R	100R	1K0	10K	100K	1M0	49	3R16	31R6	316R	3K16	31K6	316K	3M16
02	1R02	10R2	102R	1K02	10K2	102K	1M02	50	3R24	32R4	324R	3K24	32K4	324K	3M24
03	1R05	10R5	105R	1K05	10K5	105K	1M05	51	3R32	33R2	332R	3K32	33K2	332K	3M32
04	1R07	10R7	107R	1K07	10K7	107K	1M07	52	3R4	34R	340R	3K4	34K	340K	3M4
05	1R1	11R	110R	1K1	11K	110K	1M1	53	3R48	34R8	348R	3K48	34K8	348K	3M48
06	1R13	11R3	113R	1K13	11K3	113K	1M13	54	3R57	35R7	357R	3K57	35K7	357K	3M57
07	1R15	11R5	115R	1K15	11K5	115K	1M15	55	3R65	36R5	365R	3K65	36K5	365K	3M65
08	1R18	11R8	118R	1K18	11K8	118K	1M18	56	3R74	37R4	374R	3K74	37K4	374K	3M74
09	1R21	12R1	121R	1K21	12K1	121K	1M21	57	3R83	38R3	383R	3K83	38K3	383K	3M83
10	1R24	12R4	124R	1K24	12K4	124K	1M24	58	3R92	39R2	392R	3K92	39K2	392K	3M92
11	1R27	12R7	127R	1K27	12K7	127K	1M27	59	4R02	40R2	402R	4K02	40K2	402K	4M02
12	1R3	13R	130R	1K3	13K	130K	1M3	60	4R12	41R2	412R	4K12	41K2	412K	4M12
13	1R33	13R3	133R	1K33	13K3	133K	1M33	61	4R22	42R2	422R	4K22	42K2	422K	4M22
14	1R37	13R7	137R	1K37	13K7	137K	1M37	62	4R32	43R2	432R	4K32	43K2	432K	4M32
15	1R4	14R	140R	1K4	14K	140K	1M4	63	4R42	44R2	442R	4K42	44K2	442K	4M42
16	1R43	14R3	143R	1K43	14K3	143K	1M43	64	4R53	45R3	453R	4K53	45K3	453K	4M53
17	1R47	14R7	147R	1K47	14K7	147K	1M47	65	4R64	46R4	464R	4K64	46K4	464K	4M64
18	1R5	15R	150R	1K5	15K	150K	1M5	66	4R75	47R5	475R	4K75	47K5	475K	4M75
19	1R54	15R4	154R	1K54	15K4	154K	1M54	67	4R87	48R7	487R	4K87	48K7	487K	4M87
20	1R58	15R8	158R	1K58	15K8	158K	1M58	68	4R99	49R9	499R	4K99	49K9	499K	4M99
21	1R62	16R2	162R	1K62	16K2	162K	1M62	69	5R11	51R1	511R	5K11	51K1	511K	5M11
22	1R65	16R5	165R	1K65	16K5	165K	1M65	70	5R23	52R3	523R	5K23	52K3	523K	5M23
23	1R69	16R9	169R	1K69	16K9	169K	1M69	71	5R36	53R6	536R	5K36	53K6	536K	5M36
24	1R74	17R4	174R	1K74	17K4	174K	1M74	72	5R49	54R9	549R	5K49	54K9	549K	5M49
25	1R78	17R8	178R	1K78	17K8	178K	1M78	73	5R62	56R2	562R	5K62	56K2	562K	5M62
26	1R82	18R2	182R	1K82	18K2	182K	1M82	74	5R76	57R6	576R	5K76	57K6	576K	5M76
27	1R87	18R7	187R	1K87	18K7	187K	1M87	75	5R9	59R	590R	5K9	59K	590K	5M9
28	1R91	19R1	191R	1K91	19K1	191K	1M91	76	6R04	60R4	604R	6K04	60K4	604K	6M04
29	1R96	19R6	196R	1K96	19K6	196K	1M96	77	6R19	61R9	619R	6K19	61K9	619K	6M19
30	2R0	20R	200R	2K0	20K	200K	2M0	78	6R34	63R4	634R	6K34	63K4	634K	6M34
31	2R05	20R5	205R	2K05	20K5	205K	2M05	79	6R49	64R9	649R	6K49	64K9	649K	6M49
32	2R1	21R	210R	2K1	21K	210K	2M1	80	6R65	66R5	665R	6K65	66K5	665K	6M65
33	2R15	21R5	215R	2K15	21K5	215K	2M15	81	6R81	68R1	681R	6K81	68K1	681K	6M81
34	2R21	22R1	221R	2K21	22K1	221K	2M21	82	6R98	69R8	698R	6K98	69K8	698K	6M98
35	2R26	22R6	226R	2K26	22K6	226K	2M26	83	7R15	71R5	715R	7K15	71K5	715K	7M15
36	2R32	23R2	232R	2K32	23K2	232K	2M32	84	7R32	73R2	732R	7K32	73K2	732K	7M32
37	2R37	23R7	237R	2K37	23K7	237K	2M37	85	7R5	75R	750R	7K5	75K	750K	7M5
38	2R43	24R3	243R	2K43	24K3	243K	2M43	86	7R68	76R8	768R	7K68	76K8	768K	7M68
39	2R49	24R9	249R	2K49	24K9	249K	2M49	87	7R87	78R7	787R	7K87	78K7	787K	7M87
40	2R55	25R5	255R	2K55	25K5	255K	2M55	88	8R06	80R6	806R	8K06	80K6	806K	8M06
41	2R61	26R1	261R	2K61	26K1	261K	2M61	89	8R25	82R5	825R	8K25	82K5	825K	8M25
42	2R67	26R7	267R	2K67	26K7	267K	2M67	90	8R45	84R5	845R	8K45	84K5	845K	8M45
43	2R74	27R4	274R	2K74	27K4	274K	2M74	91	8R66	86R6	866R	8K66	86K6	866K	8M66
44	2R8	28R	280R	2K8	28K	280K	2M8	92	8R87	88R7	887R	8K87	88K7	887K	8M87
45	2R87	28R7	287R	2K87	28K7	287K	2M87	93	9R09	90R9	909R	9K09	90K9	909K	9M09
46	2R94	29R4	294R	2K94	29K4	294K	2M94	94	9R31	93R1	931R	9K31	93K1	931K	9M31
47	3R01	30R1	301R	3K01	30K1	301K	3M01	95	9R53	95R3	953R	9K53	95K3	953K	9M53
48	3R09	30R9	309R	3K09	30K9	309K	3M09	96	9R76	97R6	976R	9K76	97K6	976K	9M76

CHIP CAPACITOR MARKING CODE TABLE

(Value in picofarads for alphanumeric code)

Letter	0	1	2	3	4	5	6	7
A	1.0	10	100	1,000	10,000	100,000	1,000,000	10,000,000
B	1.1	11	110	1,100	11,000	110,000	1,100,000	11,000,000
C	1.2	12	120	1,200	12,000	120,000	1,200,000	12,000,000
D	1.3	13	130	1,300	13,000	130,000	1,300,000	13,000,000
E	1.5	15	150	1,500	15,000	150,000	1,400,000	15,000,000
F	1.6	16	160	1,600	16,000	160,000	1,600,000	16,000,000
G	1.8	18	180	1,800	18,000	180,000	1,800,000	18,000,000
H	2.0	20	200	2,000	20,000	200,000	2,000,000	20,000,000
J	2.2	22	220	2,200	22,000	220,000	2,200,000	22,000,000
K	2.4	24	240	2,400	24,000	240,000	2,400,000	24,000,000
L	2.7	27	270	2,700	27,000	270,000	2,700,000	27,000,000
M	3.0	30	300	3,000	30,000	300,000	3,000,000	30,000,000
N	3.3	33	330	3,300	33,000	330,000	3,300,000	33,000,000
P	3.6	36	360	3,600	36,000	360,000	3,600,000	36,000,000
Q	3.9	39	390	3,900	39,000	390,000	3,900,000	39,000,000
R	4.3	43	430	4,300	43,000	430,000	4,300,000	43,000,000
S	4.7	47	470	4,700	47,000	470,000	4,700,000	47,000,000
T	5.1	51	510	5,100	51,000	510,000	5,100,000	51,000,000
U	5.6	56	560	5,600	56,000	560,000	5,600,000	56,000,000
V	6.2	62	620	6,200	62,000	620,000	6,200,000	62,000,000
W	6.8	68	680	6,800	68,000	680,000	6,800,000	68,000,000
X	7.5	75	750	7,500	75,000	750,000	7,500,000	75,000,000
Y	8.2	82	820	8,200	82,000	820,000	8,200,000	82,000,000
Z	9.1	91	910	9,100	91,000	910,000	9,100,000	91,000,000
a	2.5	25	250	2,500	25,000	250,000	2,500,000	25,000,000
b	3.5	35	350	3,500	35,000	350,000	3,500,000	35,000,000
d	4.0	40	400	4,000	40,000	400,000	4,000,000	40,000,000
e	4.5	45	450	4,500	45,000	450,000	4,500,000	45,000,000
f	5.0	50	500	5,000	50,000	500,000	5,000,000	50,000,000
m	6.0	60	600	6,000	60,000	600,000	6,000,000	60,000,000
n	7.0	70	700	7,000	70,000	700,000	7,000,000	70,000,000
t	8.0	80	800	8,000	80,000	800,000	8,000,000	80,000,000
y	9.0	90	900	9,000	90,000	900,000	9,000,000	90,000,000

(Source: NovaCap Chip Marking System)

STANDARD MICROCIRCUIT CROSS REFERENCES

JAN/SMD	Source	JAN/SMD	Source	JAN/SMD	Source
M38510/00101	5430	M38510/01101	54181	M38510/06004	10506
M38510/00102	5420	M38510/01102	54182	M38510/06005	10507
M38510/00103	5410	M38510/01201	54121	M38510/06006	10509
M38510/00104	5400	M38510/01202	54122	M38510/06101	10531
M38510/00105	5404	M38510/01203	54123	M38510/06102	10631
M38510/00107	5401	M38510/01301	5492	M38510/06103	10576
M38510/00108	5405	M38510/01302	5493	M38510/06104	10535
M38510/00109	5403	M38510/01303	54160	M38510/06201	10504
M38510/00201	5472	M38510/01304	54163	M38510/06301	10524
M38510/00202	5473	M38510/01305	54162	M38510/06302	10525
M38510/00203	54107	M38510/01306	54161	M38510/07003	54S04
M38510/00205	5474	M38510/01307	5490	M38510/07004	54S05
M38510/00206	5470	M38510/01308	54192	M38510/07005	54S10
M38510/00301	5440	M38510/01309	54193	M38510/07006	54S20
M38510/00302	5437	M38510/01401	54150	M38510/07009	54S133
M38510/00303	5438	M38510/01403	54153	M38510/07010	54S134
M38510/00401	5402	M38510/01405	54157	M38510/07101	54S74
M38510/00403	5425	M38510/01406	54151	M38510/07102	54S112
M38510/00404	5427	M38510/01501	5475	M38510/07103	54S113
M38510/00501	5450	M38510/01503	54116	M38510/07105	54S174
M38510/00502	5451	M38510/01601	5408	M38510/07106	54S175
M38510/00602	5483	M38510/01602	5409	M38510/07201	54S40
M38510/00604	5480	M38510/01701	54174	M38510/07301	54S02
M38510/00701	5486	M38510/01702	54175	M38510/07402	54S64
M38510/00801	5406	M38510/01801	54170	M38510/07501	54S86
M38510/00802	5416	M38510/01901	54180	M38510/07502	54S135
M38510/00804	5417	M38510/02201	54H72	M38510/07601	54S194
M38510/00805	5426	M38510/02205	54H101	M38510/07602	54S195
M38510/00901	5495	M38510/02206	54H103	M38510/07701	54S138
M38510/00902	5496	M38510/02302	54H20	M38510/07702	54S139
M38510/00903	54164	M38510/02304	54H00	M38510/07801	54S181
M38510/00904	54165	M38510/02306	54H01	M38510/07802	54S182
M38510/00905	54194	M38510/02307	54H22	M38510/07901	54S151
M38510/00906	54195	M38510/02401	54H40	M38510/07902	54S153
M38510/01001	5442	M38510/03501	MMH0026	M38510/07903	54S157
M38510/01003	5444	M38510/04005	54H55	M38510/07905	54S251
M38510/01004	5445	M38510/06001	10501	M38510/07906	54S257
M38510/01005	54145	M38510/06002	10502	M38510/07907	54S258
M38510/01009	5449	M38510/06003	10505	M38510/07908	54S253

(CONTINUE)

JAN/SMD	Source	JAN/SMD	Source	JAN/SMD	Source
M38510/08001	54S11	M38510/30007	54LS20	M38510/31005	54LS09
M38510/08003	54S08	M38510/30009	54LS30	M38510/31101	54LS85
M38510/08101	54S140	M38510/30102	54LS74	M38510/31201	54LS83A
M38510/08201	54S85	M38510/30103	54LS112	M38510/31202	54LS283
M38510/15001	5485	M38510/30104	54LS113	M38510/31301	54LS13
M38510/15101	5413	M38510/30106	54LS174	M38510/31302	54LS14
M38510/15102	5414	M38510/30107	54LS175	M38510/31502	54LS93
M38510/15103	54132	M38510/30108	54LS107	M38510/31503	54LS160A
M38510/15201	54154	M38510/30109	54LS109	M38510/31504	54LS161A
M38510/15202	54155	M38510/30110	54LS76	M38510/31506	54LS169A
M38510/15203	54156	M38510/30201	54LS40	M38510/31507	54LS192
M38510/15301	54125	M38510/30202	54LS37	M38510/31508	54LS193
M38510/15302	54126	M38510/30203	54LS38	M38510/31509	54LS191
M38510/15501	54H08	M38510/30301	54LS02	M38510/31510	54LS92
M38510/15503	54H21	M38510/30302	54LS27	M38510/31511	54LS162A
M38510/15602	54148	M38510/30303	54LS266	M38510/31512	54LS163A
M38510/16101	5432	M38510/30401	54LS51	M38510/31601	54LS75
M38510/16301	54365	M38510/30402	54LS54	M38510/31603	54LS259
M38510/16302	54366	M38510/30501	54LS32	M38510/31604	54LS375
M38510/16303	54367	M38510/30502	54LS86	M38510/31901	54LS670
M38510/16304	54368	M38510/30604	54LS96	M38510/31902	54LS170
M38510/20101	0512	M38510/30605	54LS164	M38510/32002	54LS197
M38510/20301	82S126	M38510/30607	54LS395A	M38510/32102	54LS26
M38510/20302	82S129	M38510/30701	54LS138	M38510/32201	54LS365
M38510/20401	82S130	M38510/30702	54LS139	M38510/32202	54LS366
M38510/20402	82S131	M38510/30703	54LS42	M38510/32203	54LS367
M38510/20701	82S23	M38510/30801	54LS181	M38510/32204	54LS368
M38510/20702	82S123	M38510/30901	54LS151	M38510/32301	54LS125A
M38510/20703	82S23A	M38510/30902	54LS153	M38510/32302	54LS126
M38510/20704	82S123A	M38510/30903	54LS157	M38510/32401	54LS240
M38510/20802	82S141	M38510/30904	54LS158	M38510/32402	54LS241
M38510/20902	82S185	M38510/30905	54LS251	M38510/32403	54LS244
M38510/20904	82S181	M38510/30906	54LS257A	M38510/32404	54LS540
M38510/20910	82S185A	M38510/30907	54LS258A	M38510/32405	54LS541
M38510/21002	82S191	M38510/30908	54LS253	M38510/32501	54LS273
M38510/30001	54LS00	M38510/30909	54LS298	M38510/32502	54LS373
M38510/30003	54LS04	M38510/31001	54LS11	M38510/32503	54LS374
M38510/30004	54LS05	M38510/31003	54LS21	M38510/32504	54LS377
M38510/30005	54LS10	M38510/31004	54LS08	M38510/32601	54LS155

(CONTINUE)

JAN/SMD	Source	JAN/SMD	Source	JAN/SMD	Source
M38510/32602	54LS156	7700101	54LS93	8670302	82S123B
M38510/32701	54LS390	7700201	8080A	8670901	82S105
M38510/32702	54LS393	7700901	54LS160A	5962-8605201	8X350
M38510/32703	54LS490	7704201	54LS670	5962-8686801	8286
M38510/32801	54LS242	7705701	54LS244	5962-8686802	8287
M38510/32802	54LS243	7800901	10516	5962-8754301	82S62
M38510/32803	54LS245	7801001	54LS273	5962-8765701	1590
M38510/42001	8080A	7801101	54LS374	5962-8771101	54221
M38510/42101	8212	7801201	54LS240	5962-8778801	82S129
M38510/42201	8224	7801502	54S189	5962-8778802	82S129A
M38510/42301	8228	7801601	82S141	5962-8779201	10558
7600201	54LS157	7802601	54LS390	5962-8780301	4344
7600501	54LS138	8001802	54LS169A	5962-8852701	10564
7600701	54LS139	8001901	54LS09	5962-8855701	10541
7600801	54LS161A	8002001	54LS242	5962-8974801	54LS642
7601101	54LS153	8002002	54LS243	5962-8992501	54LS378
7601201	54LS75	8002101	54LS245	5962-9311101	10592
7601401	54LS83A	8002201	54S251	5962-9558101	54154
7601501	54LS197	8002301	54S258	5962-9558201	54155
7601601	54LS251	8002501	54LS170	5962-9665801	54LS14
7601701	54LS253	8301701	54LS154		
7601901	54LS298	8415501	54LS540		
7602001	54LS26	8415601	54LS541		
7603101	54LS42	8416101	54LS640		
7603301	54LS158	8417902	8283		
7603401	54LS163A	8417903	8282		
7603701	54LS257	8512601	54LS33		
7603801	54LS258A	8550201	8X305		
7604001	54S194	8602301	82S16		
7604301	54LS283	8670301	82S123A		

(Source: Lansdale Semiconductor Inc.)

SEMICONDUCTOR MANUFACTURER LOGOS



(Note: Logos are trademarks or registered trademarks of their respective companies.)

Sample Part Number Prefix and Suffix References

Manufacturer Device Identification	Part Number		Digital
	Prefix	Suffix	
TEXAS INSTRUMENTS 	Standard Prefix SNJ Conforms to MIL-PRF-38535 (QML)	Options Blank No Options 2 Series-Damping Resistor on Outputs 4 Level Shifter 25 25-Ω Line Driver	Prefix Device Suffix SN 74 LVC H 16 2 244 A DGG
	Temperature Range 54 Military 74 Commercial Family Blank Transistor-Transistor Logic ABT Advanced BiCMOS Technology AC/ACT Advanced CMOS Logic AHC/AHCT Advanced High-Speed CMOS Logic ALB Advanced Low Voltage BiCMOS ALS Advanced Low-Power Schottky Logic ALVC Advanced Low-Voltage CMOS Technology AS Advanced Schottky Logic BCT BiCMOS Bus-Interface Technology F F Logic HC/HCT High-Speed CMOS Logic HSTL High-Speed Transceiver Logic LS Low-Power Schottky Logic LV Low-Voltage CMOS Technology LVC Low-Voltage CMOS Technology LVT Low-Voltage BiCMOS Technology S Schottky Logic	Function 244 Non-inverting Buffer/Driver 374 D-Type Flip-Flop 573 D-Type Transparent Latch 640 Inverting Transceiver Device Revision Blank No Revision Designator A-Z	Standard Prefix Temperature Range Family Special Features Bit Width Options Function Device Revision Packages
	Special Features Blank No Special Features D Level-Shifting Diode (CBTD) H Bus Hold (ALVCH) R Damping Resistor on Inputs/Outputs (LVCR) S Schottky Clamping Diode (CBTS)	Packages D, DW Small-Outline Integrated Circuit (SOIC) DB, DL Shrink Small-Outline Package (SSOP) DBB, DGV Thin Very Small-Outline Package (TVSOP) DBQ Quarter-Size Outline Package (QSOP) DBV, DCK Small-Outline Transistor Package (SOT) DGG, PW Thin-Shrink Transistor Package (TSOP) FK Leadless Ceramic Chip Carrier (LCCC) FN Plastic Leaded Chip Carrier (PLCC) GB Ceramic Pin Grid Array (CPGA) HFP, HS, HT Ceramic Quad Flat Package (CQFP) J, JT Ceramic Dual-In-Line Package (CDIP) N, NP, NT Plastic Dual-In-Line Package (PDIP) PAG, PN, PZ Thin Quad Flat Package (TQFP) PH, PQ, RC Quad Flat Package (QFP) W, WA, WD Ceramic Flat Package (CFP)	
	Bit Width Blank Gates, MSI, and Octals 1G Single Gate 8 Octal IEEE 1149.1 (JTAG) 16 Widebus™ (16, 18, and 20 bit) 18 Widebus IEEE 1149.1 (JTAG) 32 Widebus+™ (32 and 36 bit)		

(Courtesy of Texas Instruments)

Appendix B

VISIO REFERENCES

MICROSOFT VISIO 2007 QUICK REFERENCE CARD	B-2
BASIC ELECTRICAL TEMPLATES (Fundamental Items)	B-4
BASIC ELECTRICAL TEMPLATES (Semiconductors and Switches/Relays)	B-5
CIRCUITS AND LOGIC TEMPLATES (Analog and Digital Logic)	B-6
CIRCUITS AND LOGIC TEMPLATES (Integrated Circuit Components)	B-7

Microsoft® Visio 2007 Quick Reference Card



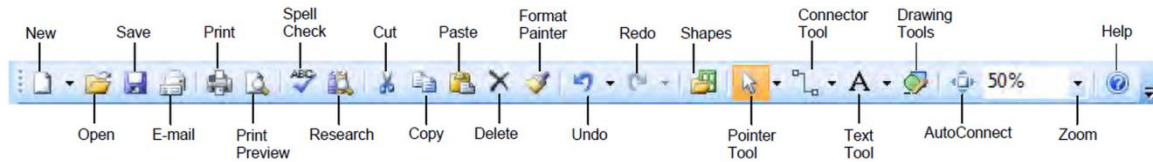
CustomGuide
Your Organization's Personal Trainer

888.903.2432 | www.customguide.com

Visio 2007 Workspace	Keyboard Shortcuts																																																																				
	<table border="0"> <tr><td>Actual size</td><td>Ctrl + Shift + I</td></tr> <tr><td>Align Shapes</td><td>F8</td></tr> <tr><td>Bring to Front</td><td>Ctrl + Shift + F</td></tr> <tr><td>Send to Back</td><td>Ctrl + Shift + B</td></tr> <tr><td>Cascade</td><td>Alt + F7</td></tr> <tr><td>Cut</td><td>Ctrl + X</td></tr> <tr><td>Copy</td><td>Ctrl + C</td></tr> <tr><td>Duplicate</td><td>Ctrl + D</td></tr> <tr><td>Field</td><td>Ctrl + F9</td></tr> <tr><td>Fill</td><td>F3</td></tr> <tr><td>Flip Horizontal</td><td>Ctrl + H</td></tr> <tr><td>Flip Vertical</td><td>Ctrl + J</td></tr> <tr><td>Toggle Glue</td><td>F9</td></tr> <tr><td>Group</td><td>Ctrl + G</td></tr> <tr><td>Ungroup</td><td>Ctrl + Shift + U</td></tr> <tr><td>Display Help</td><td>F1</td></tr> <tr><td>Insert Hyperlink</td><td>Ctrl + K</td></tr> <tr><td>Line</td><td>Shift + F3</td></tr> <tr><td>Macros</td><td>Alt + F8</td></tr> <tr><td>New Drawing</td><td>Ctrl + N</td></tr> <tr><td>Open</td><td>Ctrl + O</td></tr> <tr><td>Close File</td><td>Ctrl + F4</td></tr> <tr><td>Full Screen</td><td>F5</td></tr> <tr><td>Print Preview</td><td>Ctrl + F2</td></tr> <tr><td>Redo</td><td>Ctrl + Y</td></tr> <tr><td>Undo</td><td>Ctrl + Z</td></tr> <tr><td>Repeat</td><td>F4</td></tr> <tr><td>Find</td><td>Ctrl + F</td></tr> <tr><td>Rotate Left</td><td>Ctrl + L</td></tr> <tr><td>Rotate Right</td><td>Ctrl + R</td></tr> <tr><td>Save</td><td>Ctrl + S</td></tr> <tr><td>Toggle Snap</td><td>Shift + F9</td></tr> <tr><td>Snap and Glue</td><td>Alt + F9</td></tr> <tr><td>Spelling</td><td>F7</td></tr> </table>	Actual size	Ctrl + Shift + I	Align Shapes	F8	Bring to Front	Ctrl + Shift + F	Send to Back	Ctrl + Shift + B	Cascade	Alt + F7	Cut	Ctrl + X	Copy	Ctrl + C	Duplicate	Ctrl + D	Field	Ctrl + F9	Fill	F3	Flip Horizontal	Ctrl + H	Flip Vertical	Ctrl + J	Toggle Glue	F9	Group	Ctrl + G	Ungroup	Ctrl + Shift + U	Display Help	F1	Insert Hyperlink	Ctrl + K	Line	Shift + F3	Macros	Alt + F8	New Drawing	Ctrl + N	Open	Ctrl + O	Close File	Ctrl + F4	Full Screen	F5	Print Preview	Ctrl + F2	Redo	Ctrl + Y	Undo	Ctrl + Z	Repeat	F4	Find	Ctrl + F	Rotate Left	Ctrl + L	Rotate Right	Ctrl + R	Save	Ctrl + S	Toggle Snap	Shift + F9	Snap and Glue	Alt + F9	Spelling	F7
Actual size	Ctrl + Shift + I																																																																				
Align Shapes	F8																																																																				
Bring to Front	Ctrl + Shift + F																																																																				
Send to Back	Ctrl + Shift + B																																																																				
Cascade	Alt + F7																																																																				
Cut	Ctrl + X																																																																				
Copy	Ctrl + C																																																																				
Duplicate	Ctrl + D																																																																				
Field	Ctrl + F9																																																																				
Fill	F3																																																																				
Flip Horizontal	Ctrl + H																																																																				
Flip Vertical	Ctrl + J																																																																				
Toggle Glue	F9																																																																				
Group	Ctrl + G																																																																				
Ungroup	Ctrl + Shift + U																																																																				
Display Help	F1																																																																				
Insert Hyperlink	Ctrl + K																																																																				
Line	Shift + F3																																																																				
Macros	Alt + F8																																																																				
New Drawing	Ctrl + N																																																																				
Open	Ctrl + O																																																																				
Close File	Ctrl + F4																																																																				
Full Screen	F5																																																																				
Print Preview	Ctrl + F2																																																																				
Redo	Ctrl + Y																																																																				
Undo	Ctrl + Z																																																																				
Repeat	F4																																																																				
Find	Ctrl + F																																																																				
Rotate Left	Ctrl + L																																																																				
Rotate Right	Ctrl + R																																																																				
Save	Ctrl + S																																																																				
Toggle Snap	Shift + F9																																																																				
Snap and Glue	Alt + F9																																																																				
Spelling	F7																																																																				
The Getting Started Window																																																																					
- New in Visio 2007 is the Getting Started windows, which appears every time you open the application. Here you can browse template categories, access the Recent Templates list, and preview sample diagrams (Pro edition only). - The Getting Started window appears by default every time you open the application. You can also access it using File → New → Getting Started.																																																																					
Diagram Templates																																																																					
- There are eight template categories and over 50 diagram templates. - A template contains everything you need to create a specific type of drawing. Each template comes with its own set of stencils, shapes, and menus. Business: Organize cause and effect, finances, workflow and more. Templates: Workflow Diagram, Brainstorming, Pivot Diagram. Maps and Floor Plans: Use to create directional maps, floor plans, building plans... Templates: Maps, Floor Plan, Security, HVAC Plan, Access Plan. Software and Database: Software structure and system document, design and produce databases. Templates: Database Models, COM and OLE, Express-G. General: Basic flowcharts and diagrams. Templates: Basic Diagram, Flowchart, Block Diagram Engineering: Create electrical, pneumatic, hydraulic diagrams and instructions. Templates: Basic Electrical, Part Assembly, Processes. Network: Organizational web site, network, directory... Templates: Network Diagram, Conceptual Web Site, Rack Diagram. Schedule: Calendars, timelines and progress charts. Templates: Calendar, Gantt and PERT Charts, Timeline. Flowchart: Visual diagram of processes. Templates: Basic Flowchart Cross-Functional, Data Flow Diagram.	Formatting Text <table border="0"> <tr><td>Bold</td><td>Ctrl + B</td></tr> <tr><td>Italics</td><td>Ctrl + I</td></tr> <tr><td>Underline</td><td>Ctrl + U</td></tr> <tr><td>Double Underline</td><td>Ctrl + Shift + D</td></tr> <tr><td>Small Caps</td><td>Ctrl + Shift + K</td></tr> </table>	Bold	Ctrl + B	Italics	Ctrl + I	Underline	Ctrl + U	Double Underline	Ctrl + Shift + D	Small Caps	Ctrl + Shift + K																																																										
Bold	Ctrl + B																																																																				
Italics	Ctrl + I																																																																				
Underline	Ctrl + U																																																																				
Double Underline	Ctrl + Shift + D																																																																				
Small Caps	Ctrl + Shift + K																																																																				

The Fundamentals

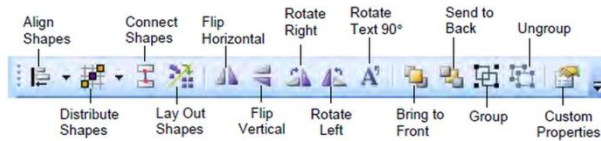
The Standard Toolbar



- **Creating a New Diagram from a Template:** Select **File** → **New** from menu, select a template category, and select a template. Or click the **New** button list arrow, select a category and then select a template.
- **Creating a New Diagram from Scratch:** Click the **New** button, or press **Ctrl + N**.
- **Opening a Diagram:** Click the **Open** button, or select **File** → **Open** from menu, or press **Ctrl + O**.
- **Saving a Diagram:** Click the **Save** button, or select **File** → **Save** from menu, or press **Ctrl + S**.
- **Printing a Diagram:** Click the **Print** button, or select **File** → **Print** from menu, or press **Ctrl + P**.
- **Viewing a Diagram** (Full screen mode): Press **F5**. Use the mouse or arrow keys to navigate between pages.
- **Changing Page Orientation:** Select **File** → **Page Setup** from menu and select Landscape or Portrait.
- **Finding a Shape:** Type what you're looking for in the **Search for shapes** box in the **Shapes** window and click the **Search** button.
- **Selecting a Shape:** Click it.

Working with Shapes

The Action Toolbar



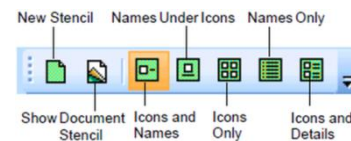
- **Inserting a Shape:** Click and drag the shape from the stencil to the desired location in the drawing page.
- **Deleting a Shape:** Select the shape(s). Press **Delete**.
- **Adding text to a Shape:** Click shape and start typing.
- **Moving a Shape:** Click shape and drag to new location.
- **Resizing a Shape:** Select shape and drag one of its handles.
- **Aligning or Distributing Shapes:** Select the shapes and click **Align Shapes** or **Distribute Shapes** button, or select **Shapes** → **Align Shapes** or **Distribute Shapes** from menu. Select required option and click OK.
- **Formatting Shapes:** Select the shape, click **Fill Color** or **Line** button list arrow on Formatting toolbar and select an option from the list, or select **Format** → **Fill** or **Line** from menu.
- **Grouping Shapes:** Select the shapes and click **Group** button, or select **Shapes** → **Grouping** → **Group** from menu, or press **Ctrl + G**.

Themes

- **Applying a Theme:**
Select **Format** → **Theme** from the menu, or click the **Theme** button on the Formatting toolbar. The Theme - Colors task pane appears; click **Theme Effects** to open the Theme - Effects task pane.

Stencils

The Stencil Toolbar

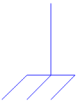
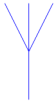
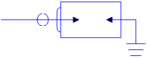


- A stencil is a collection of shapes related to a template you are currently working on. Each template has a different set of stencils assigned to it.
- **Opening a Stencil:** Click the **Shapes** button, or select **File** → **Shapes** → **Open Stencil** from menu.
- **Moving a Stencil:** Click and drag the stencil by its title bar to a new location in the program window.
- **Removing a Stencil:** Right-click the stencil and select **Close** from the shortcut menu.
- **Creating a New Stencil:** Select **File** → **Shapes** → **New Stencil** from menu and add shapes to the stencil.
- **Changing how Shapes are displayed:** Click one of the five view buttons on the Stencil toolbar above.

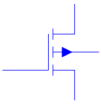
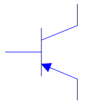
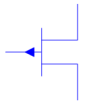
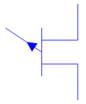
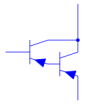
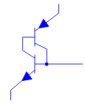
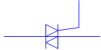
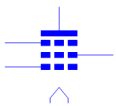
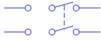
Connecting Shapes

- **Using AutoConnect:** A new feature in Visio 2007, lets you add, connect, distribute and align a whole series of shapes with only a few clicks. To turn on feature, click the **AutoConnect** button. To use it, click a shape in the Shapes window and then rest the mouse cursor over the shape on the drawing page you want to connect the new shape to. Click the **blue connection arrow** on the side of the shape you want to connect to. Visio automatically adds and connects the shape from the Shapes window, align and distribute the shapes as necessary.

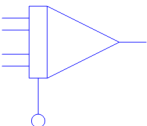
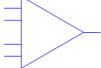
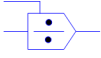
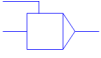
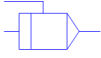
BASIC ELECTRICAL TEMPLATES
(Fundamental Items)

 Ground	 Equipotentiality	 Chassis	 Battery	 Resistor	 Attenuator
 Capacitor	 Antenna	 Loop Antenna	 Crystal	 Circuit Breaker	 Fuse
 Transducer	 Transducer 2	 Pickup Head	 Inductor	 Half Inductor	 Pulse
 Alternating Pusle	 Sawtooth	 Step Function	 Indicator	 Material	 Delay Element
 Surge Protector	 Surge Protector 2	 Magnet Core	 Ferrite Core	 Igniter Plug	 Bell
 Buzzer	 Thermal Element	 Thermocouple	 Thermopile	 Lamp	 Lamp 2
 Fluorescent Lamp	 Speaker	 Microphone 2	 Oscillator	 AC Source	 DC Source

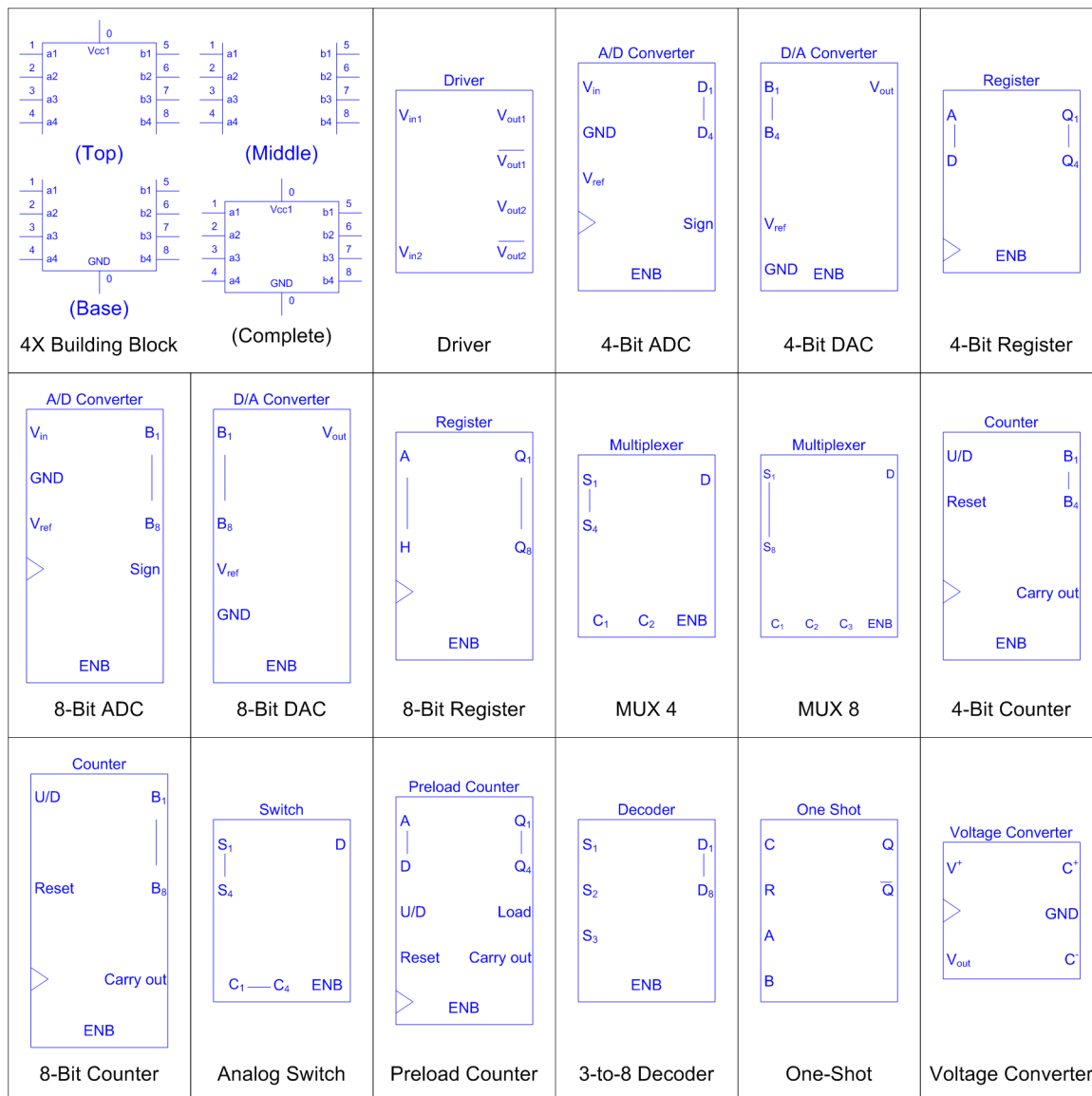
BASIC ELECTRICAL TEMPLATES (Semiconductors and Switches/Relays)

					
MOSFET	Bipolar	Junction	Unijunction	Darlington	Latch
					
IGFET N-Type	IGFET P-Type	Diode	Tunnel Diode	Backward Diode	Varactor
					
Diac	Triac	Controlled Switch	Controlled Rectifier	Photodiode	Breakdown Diode
					
Tube Diode	Tube Triode	Tube Tetrode	Tube Pentode	Turn-off Triode	
					
SPST	SPDT	DPST	DPDT	Make Contact	Break Contact
					
Two Way Contact	Circuit Breaker	2-Position Switch	3-Position Switch	4-Position Switch	Manual Switch
					
Pushbutton Make	Pushbutton Break	Selector Switch	Fuse	Relay Coil	Relay Contact

CIRCUITS AND LOGIC TEMPLATES
(Analog and Digital Logic)

 Inverter	 Buffer	33 MHz  Clock	 Function Generator	 Amplifier	 Converter
 Logic Gate 1	 Logic Gate 2	 Flip-Flop	 Negative Logic Dot	 Function Pot	 Positional Servo
 Delay Element	 I/O Port	 Signal Waveforms	 3-State Buffer	 Integrator	 Summing Amplifier
 Multiplier	 Divider	 Function Gen 2	 Generic Integrator	 Op-Amp	 Op-Amp 2

CIRCUITS AND LOGIC TEMPLATES (Integrated Circuit Components)

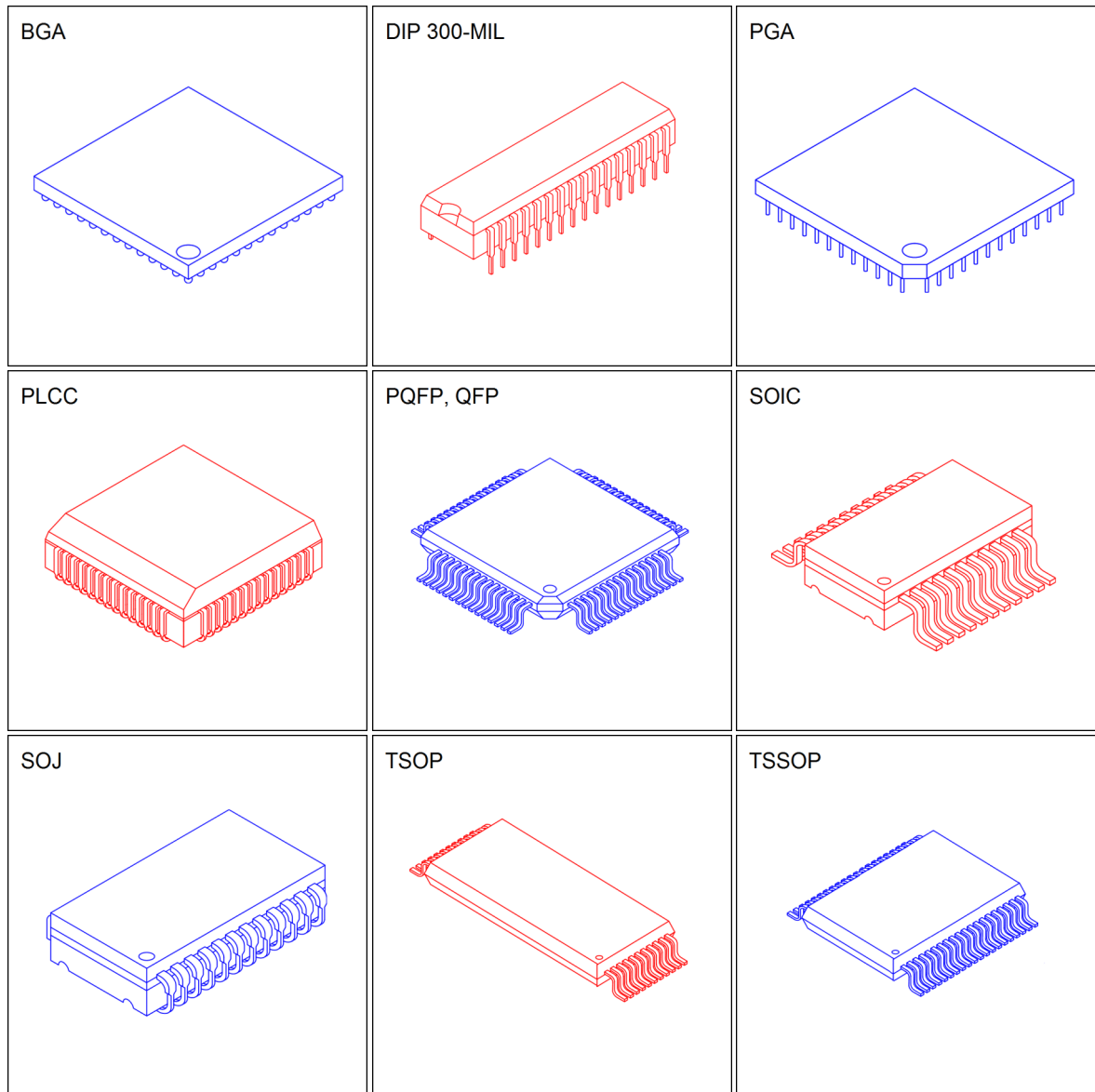


Appendix C

PCB LAYOUT

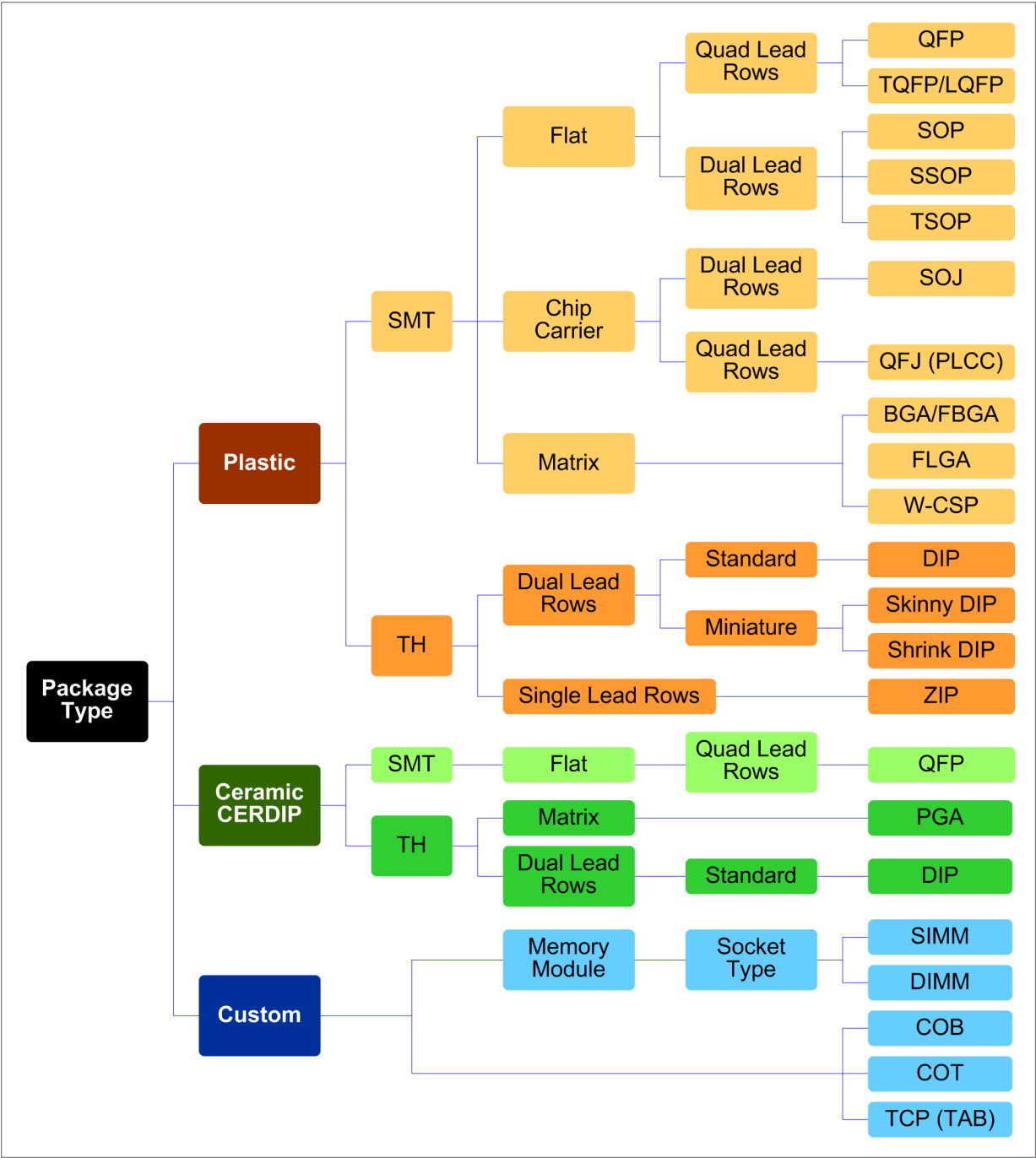
INTEGRATED CIRCUIT PACKAGES	C-1
IC PACKAGE CLASSIFICATIONS	C-2
IC PACKAGE ABBREVIATIONS	C-3
BALL GRID ARRAY (BGA)	C-4
DUAL IN-LINE PACKAGE (DIP)	C-5
PLASTIC LEADED CHIP CARRIER (PLCC)	C-6
QUAD FLAT L-LEADED PACKAGE (QFP)	C-7
SMALL OUTLINE J-LEADED PACKAGE	C-8
SMALL OUTLINE L-LEADED PACKAGE	C-9
TYPES OF SCSI CONNECTORS	C-10
TRANSISTOR PACKAGES AND PINOUTS	C-11

INTEGRATED CIRCUIT PACKAGES



(Source: Wikipedia)

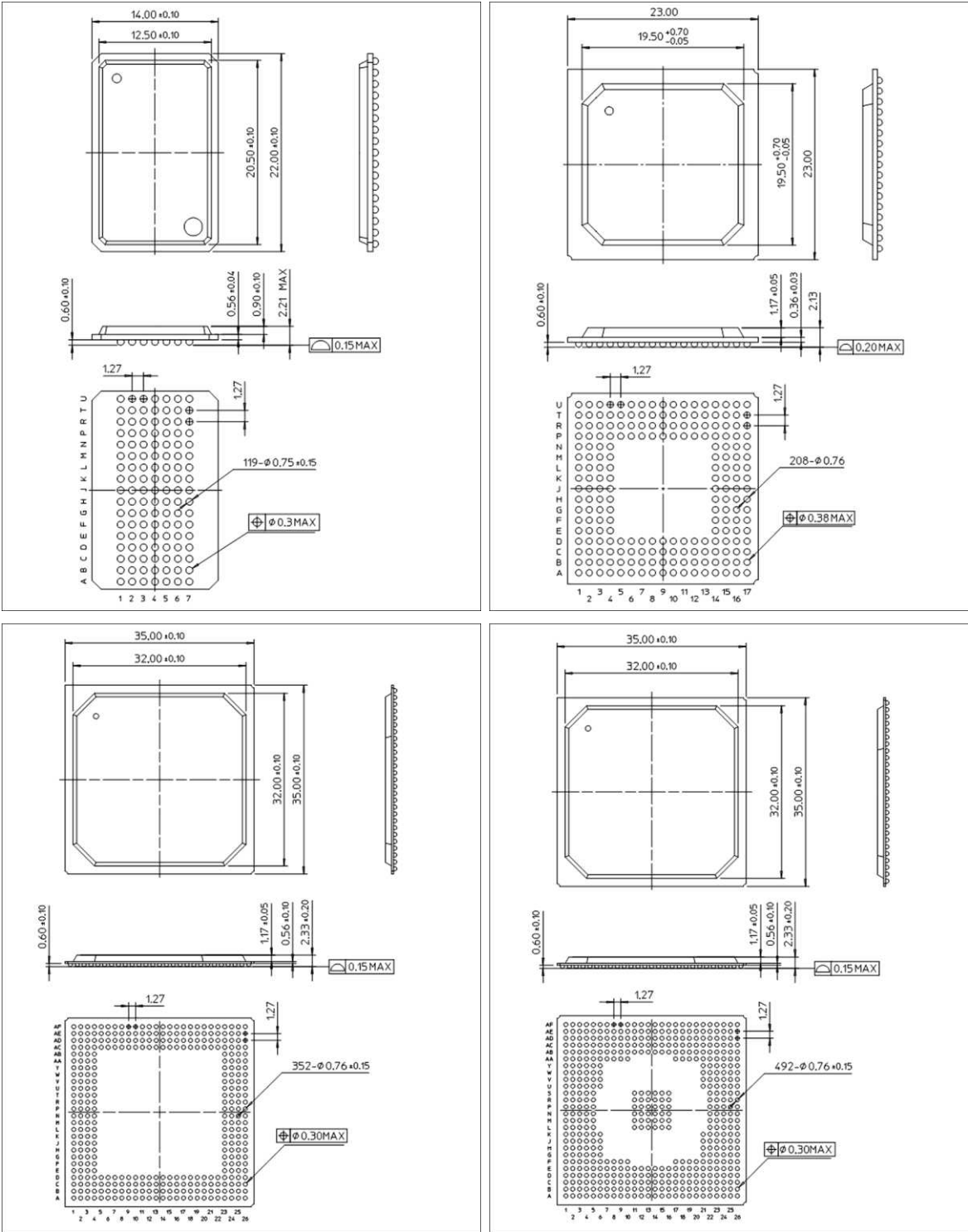
IC PACKAGE CLASSIFICATIONS



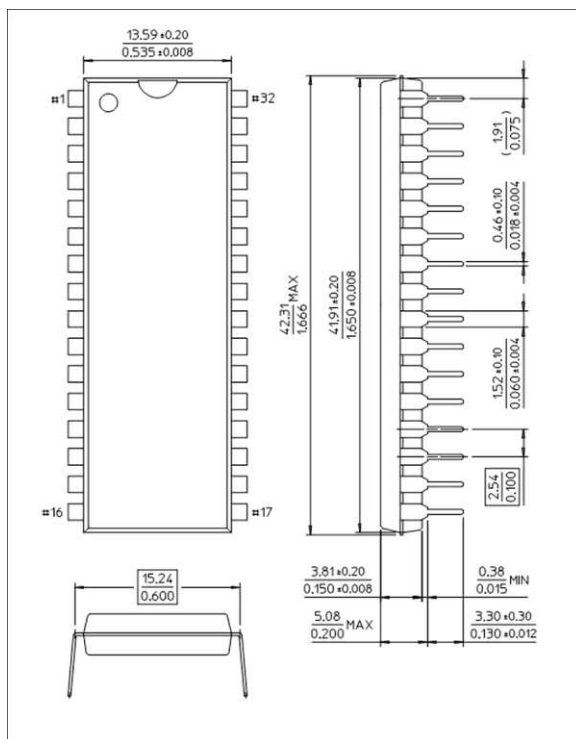
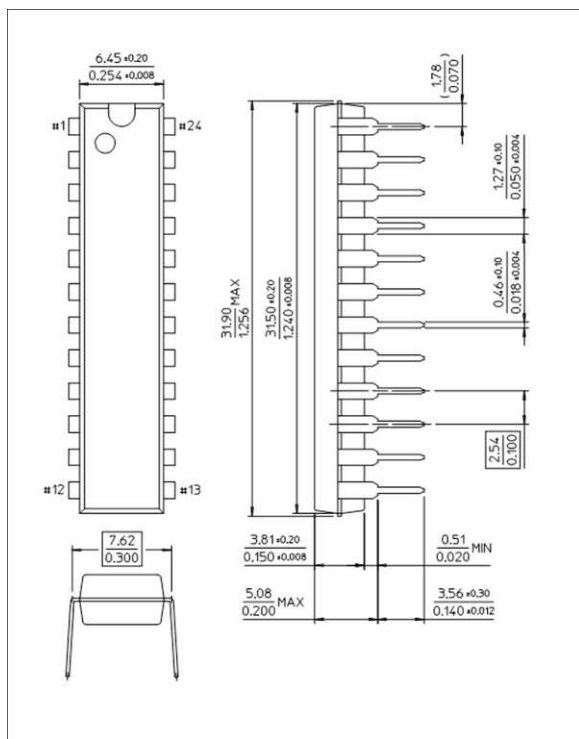
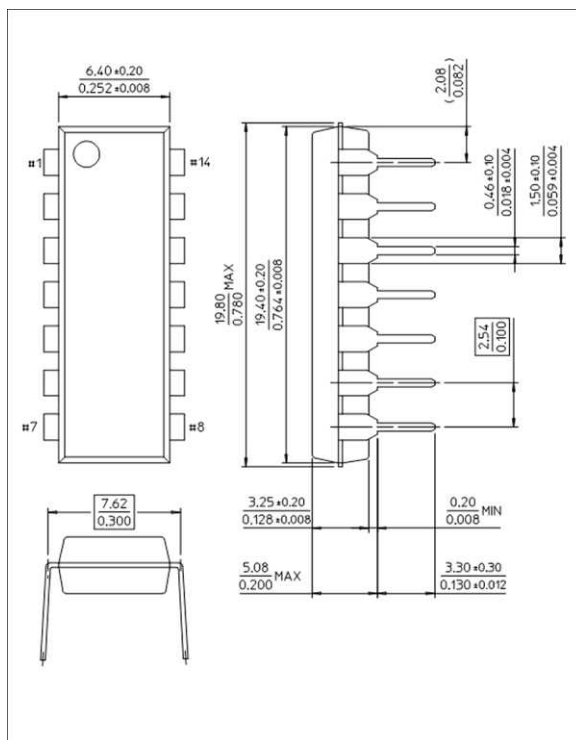
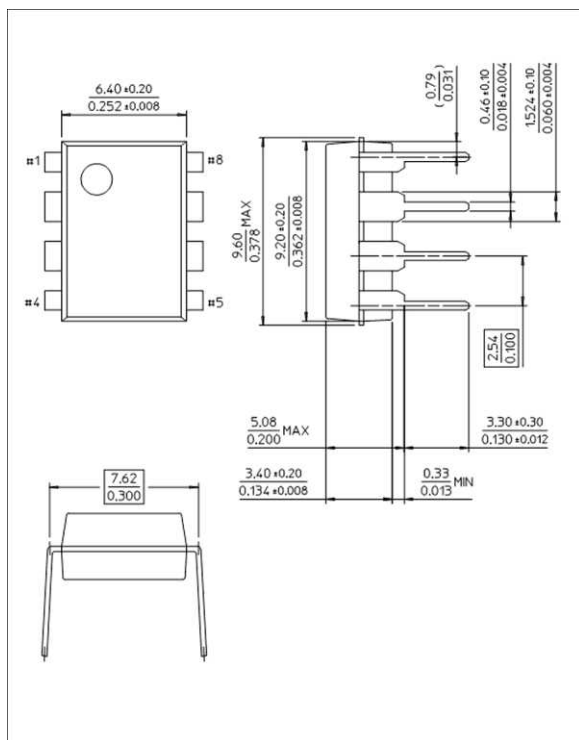
IC PACKAGE ABBREVIATIONS

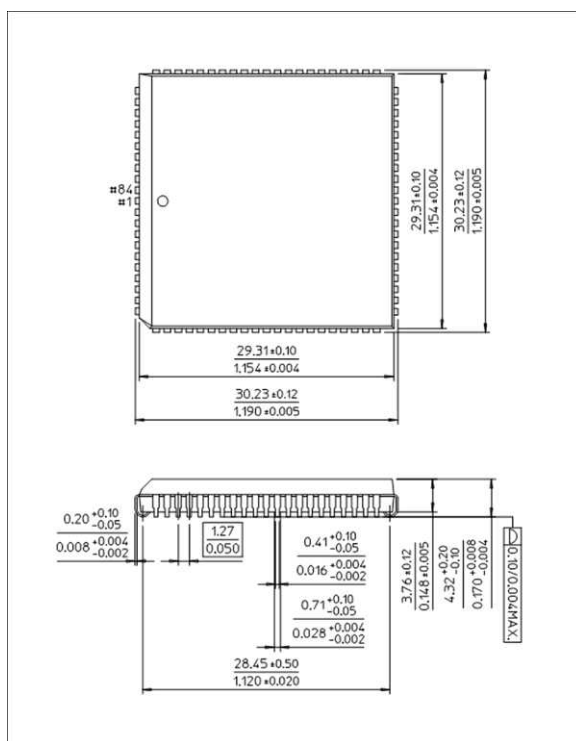
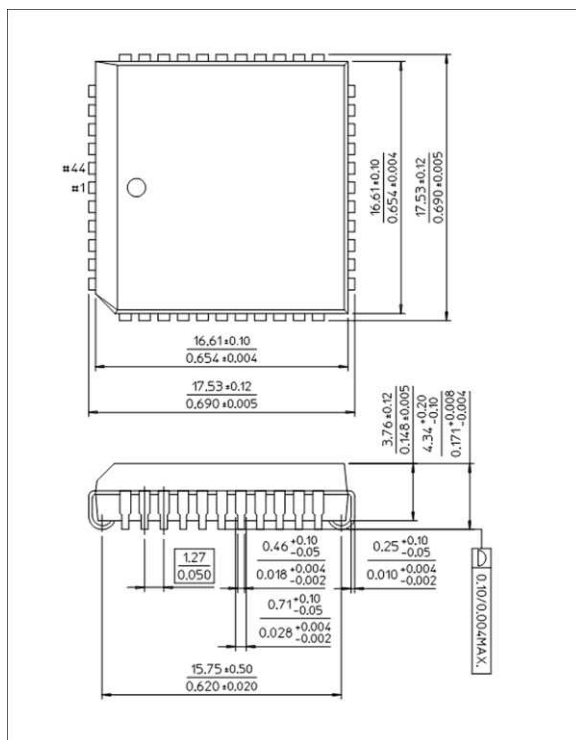
BGA	Ball Grid Array
COB	Chip On Board
COT	Chip On Tape
DIMM	Dual In-line Memory Module
DIP	Dual In-line Package
FLGA	Fine-Pitch Land Grid Array
QFJ	Quad Flat J-Leaded Package
QFP	Quad Flat L-Leaded Package
LQFP	Low Profile Quad Flat L-Leaded Package
PLCC	Plastic Leaded Chip Carrier
PGA	Pin Grid Array
SIMM	Single In-line Memory Module
SMT	Surface Mount Technology
SOJ	Small Outline J-Leaded Package
SOP	Small Outline L-Leaded Package
SSOP	Shrink Small Outline L-Leaded Package
TAB	Tape Automated Bonding
TCP	Tape Carrier Package
TH	Through-Hole
TQFP	Thin Quad Flat L-Leaded Package
TSOP	Thin Small Outline L-Leaded Package
W-CSP	Wafer Level Chip Size Package
ZIP	Zigzag In-line Package

BALL GRID ARRAY (BGA)

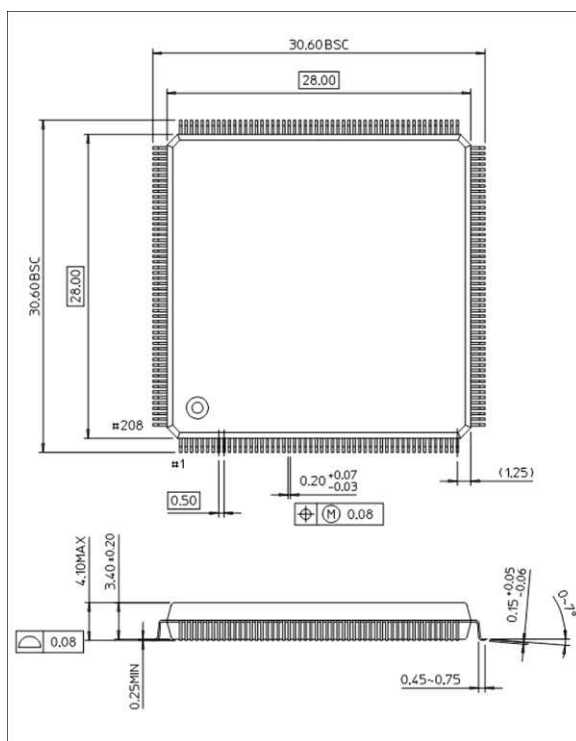
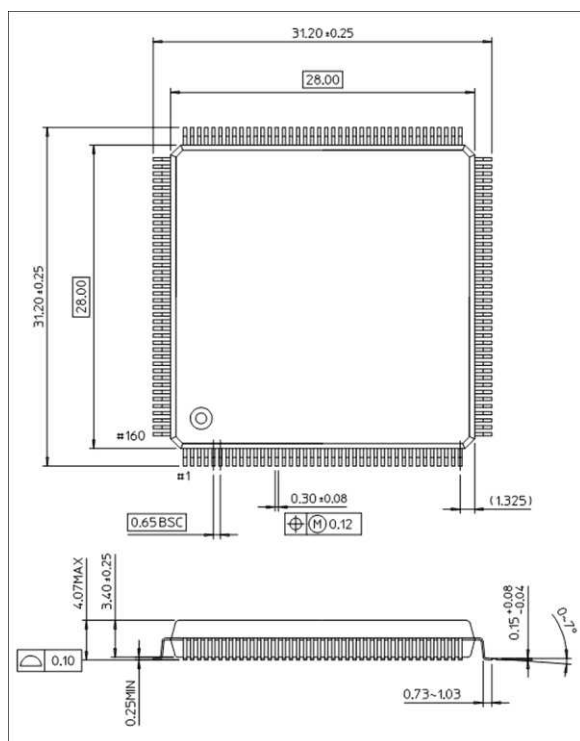
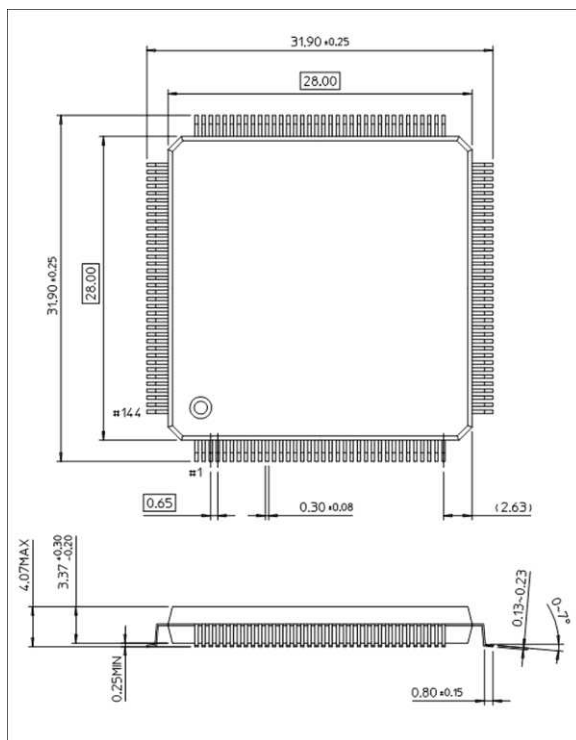
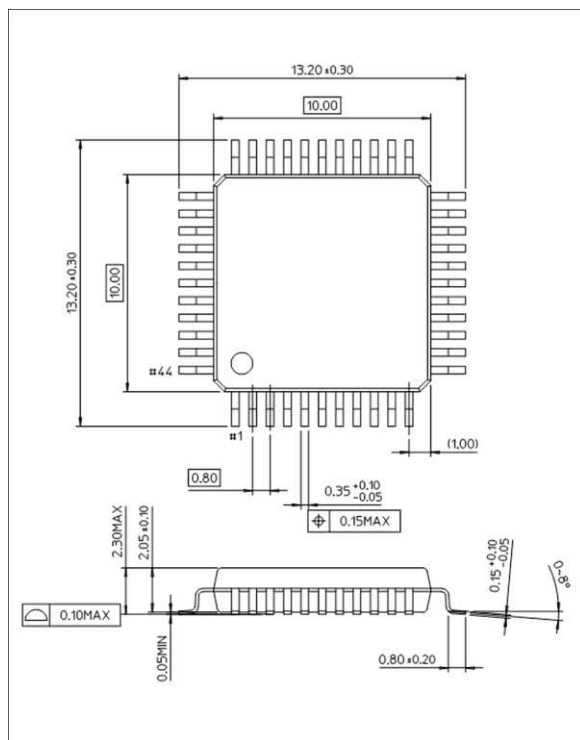


DUAL IN-LINE PACKAGE (DIP)

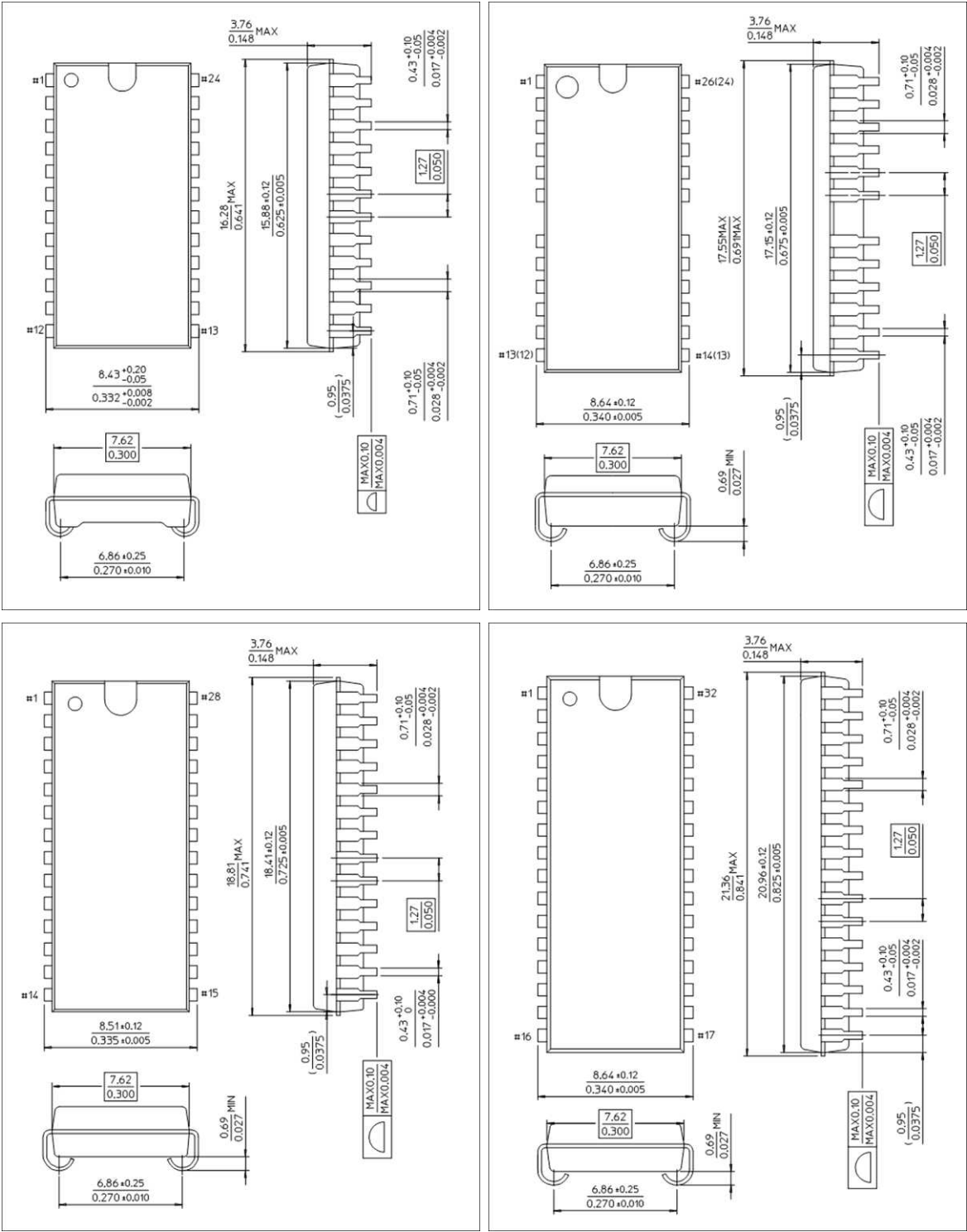




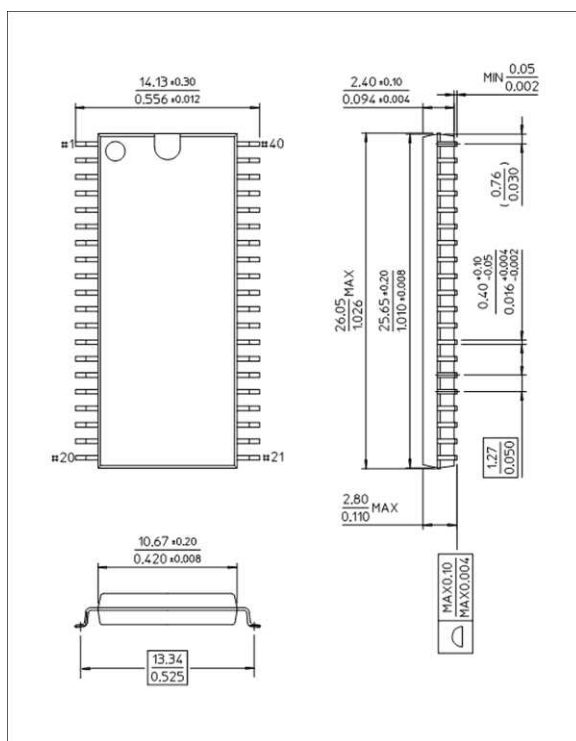
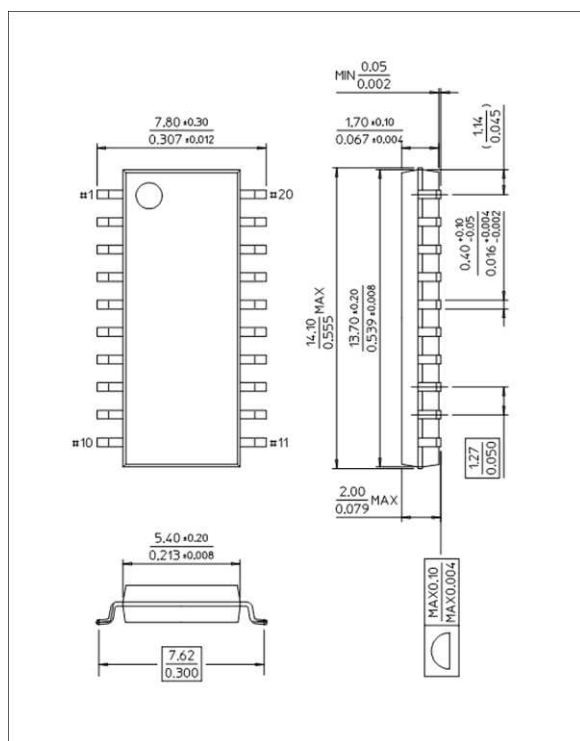
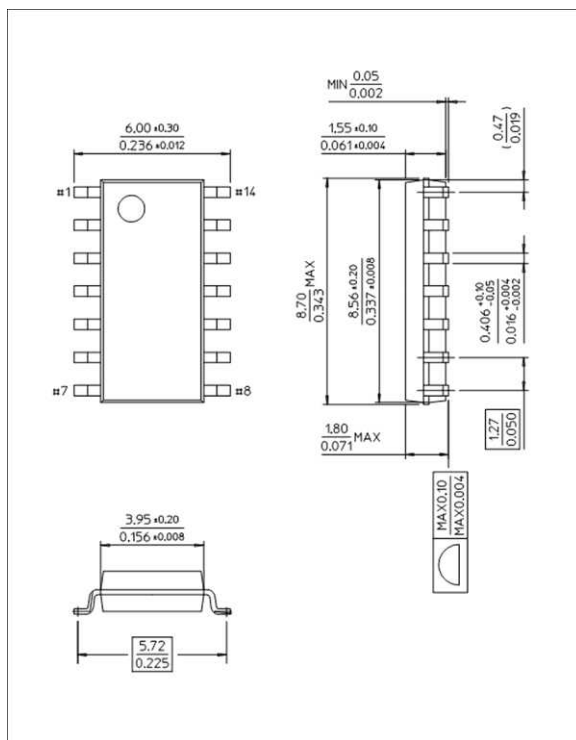
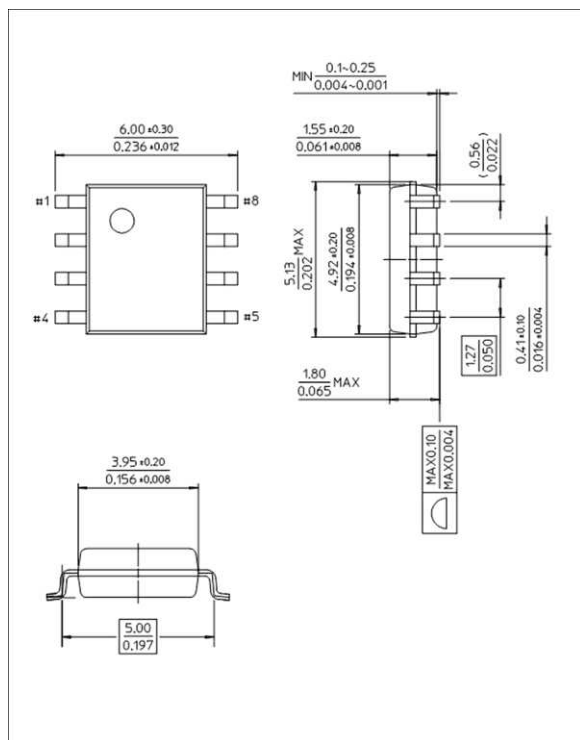
QUAD FLAT L-LEADED PACKAGE (QFP)



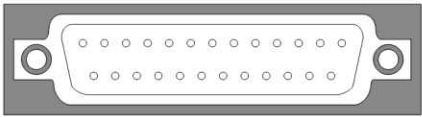
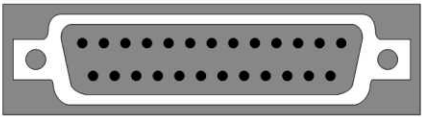
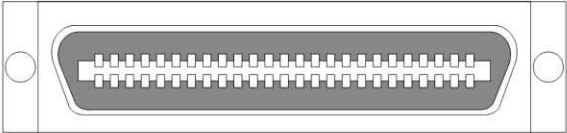
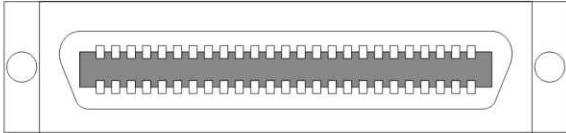
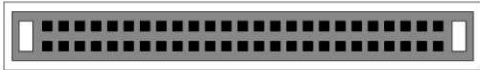
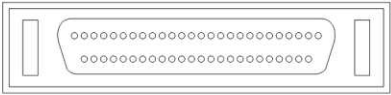
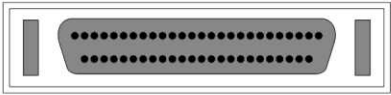
SMALL OUTLINE J-LEADED PACKAGE (SOJ)



SMALL OUTLINE L-LEADED PACKAGE (SOP)



TYPES OF SCSI CONNECTORS

DB-25, Male External 	DB-25, Female External 
Low-Density, 50-Way, Male External 	Low-Density, 50-Way, Female External 
Low-Density, 50-Way, Male Internal 	Low-Density, 50-Way, Female Internal 
High-Density, 50-Way, Male External 	High-Density, 50-Way, Female External 
High-Density, 68-Way, Male External 	High-Density, 68-Way, Female Internal 
High-Density, 68-Way, Male Internal 	High-Density, 68-Way, Female Internal 

Adaptec Terminology	Alternative Terminology
Low-density 50-pin	Centronics 50-pin
High-density 50-pin	Micro DB-50 or Mini DB-50
High-density 68-pin	Micro DB-68 or Mini DB-68
Very high-density condensed 68-pin	Ultra Micro DB-68

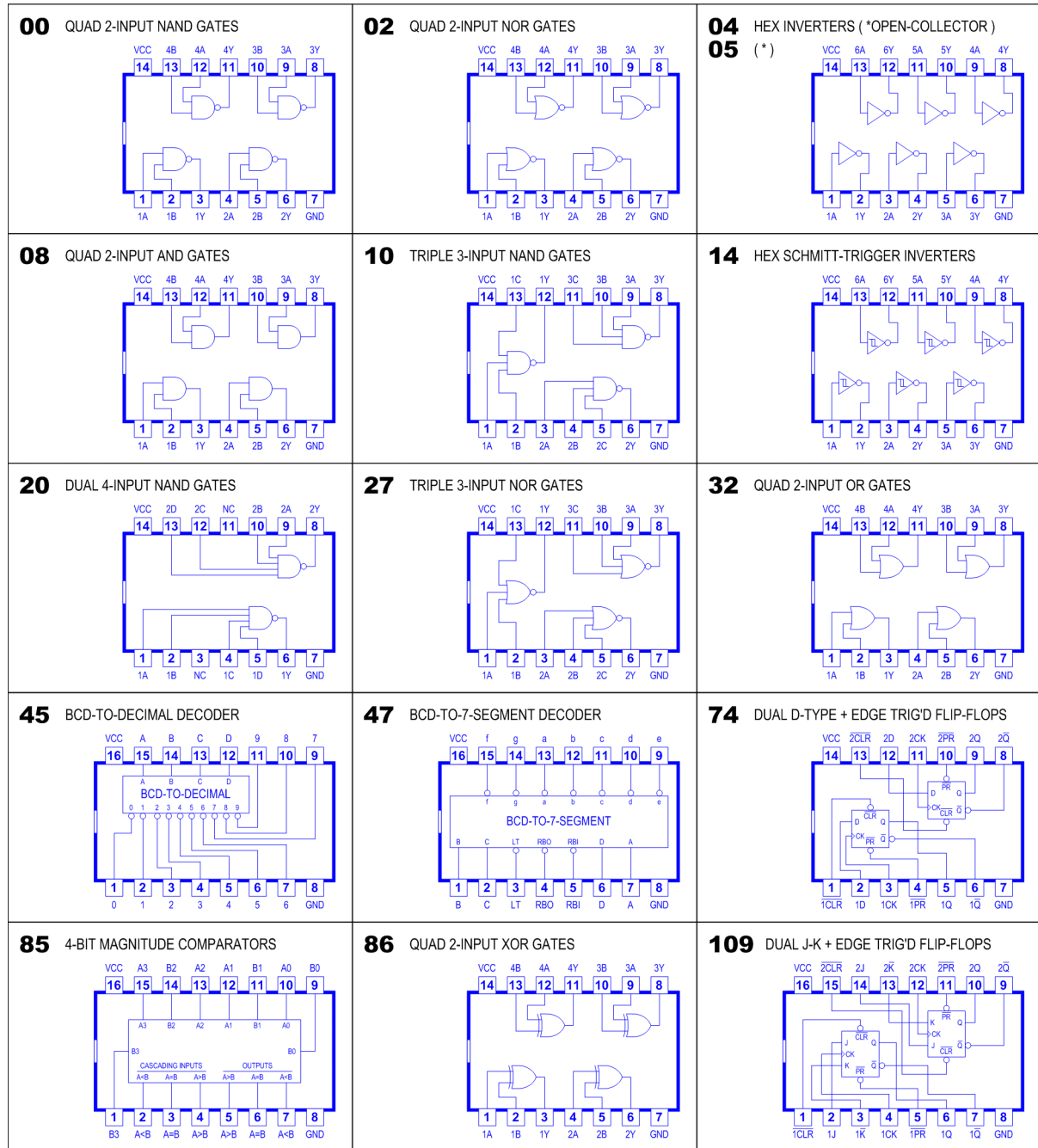
(Source: Unknown)

Appendix D

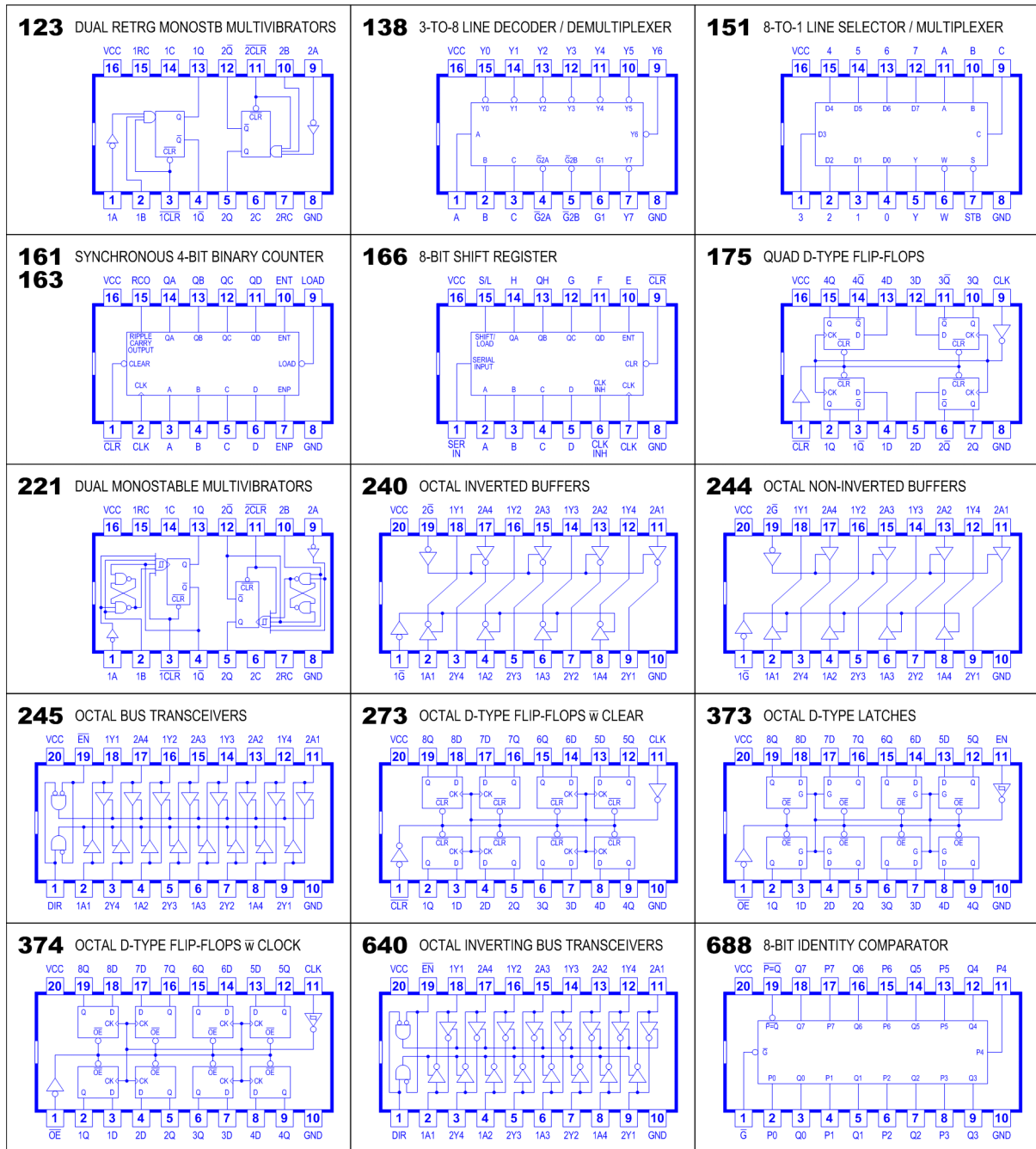
SCHEMATIC ENTRY

COMMON 54 / 74-SERIES TTL IC PINOUTS	D-1
COMMON 4000-SERIES CMOS IC PINOUTS	D-3
BASIC OP AMP CONFIGURATIONS	D-5
SWITCH-MODE POWER SUPPLY TOPOLOGIES	D-6
TYPES OF PASSIVE FILTER CIRCUITS	D-7
TYPES OF ACTIVE FILTER CIRCUITS	D-8

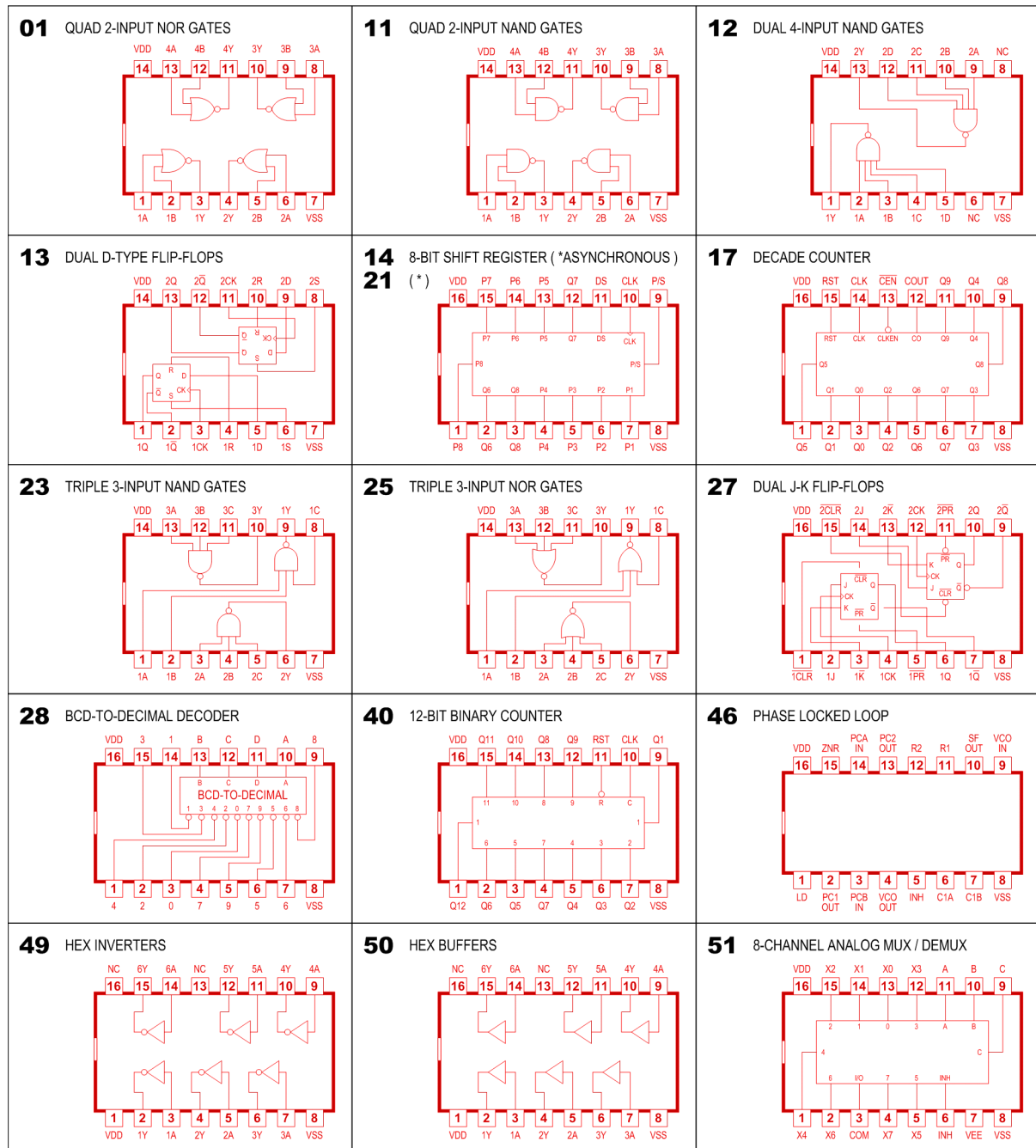
COMMON 54/74-SERIES TTL IC PINOUTS



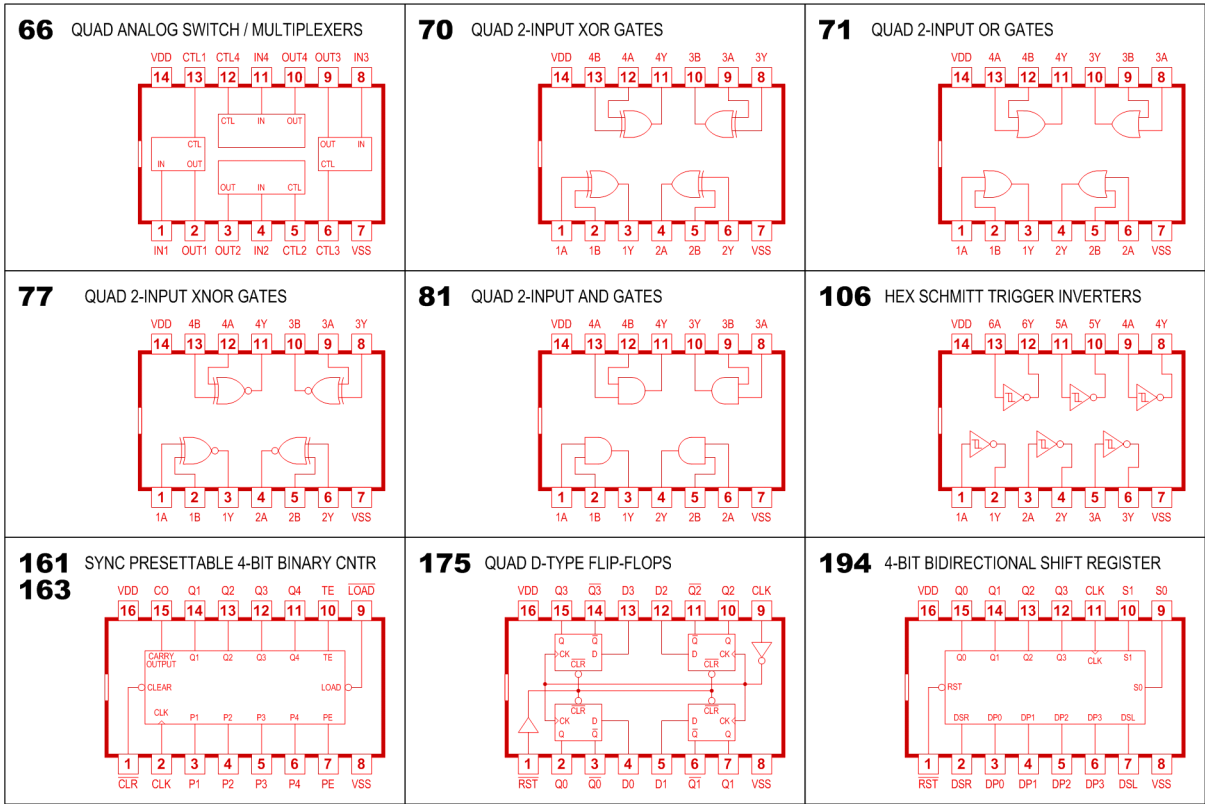
(CONTINUE)



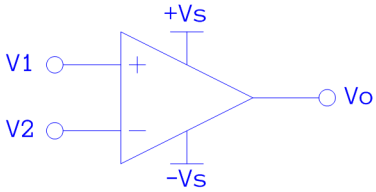
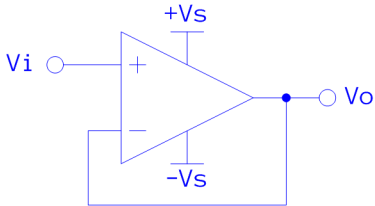
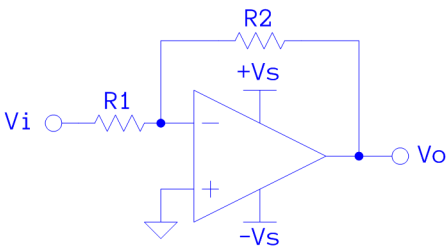
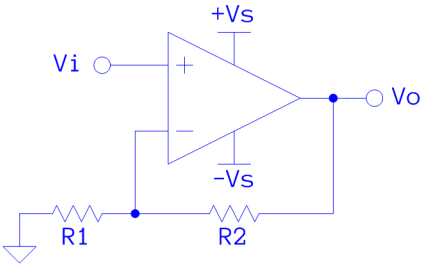
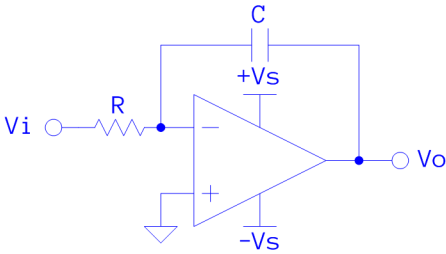
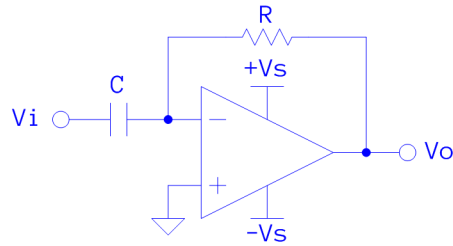
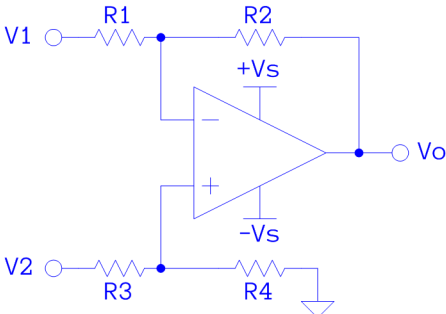
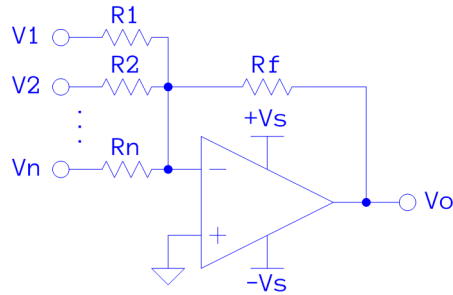
COMMON 4000-SERIES CMOS IC PINOUTS



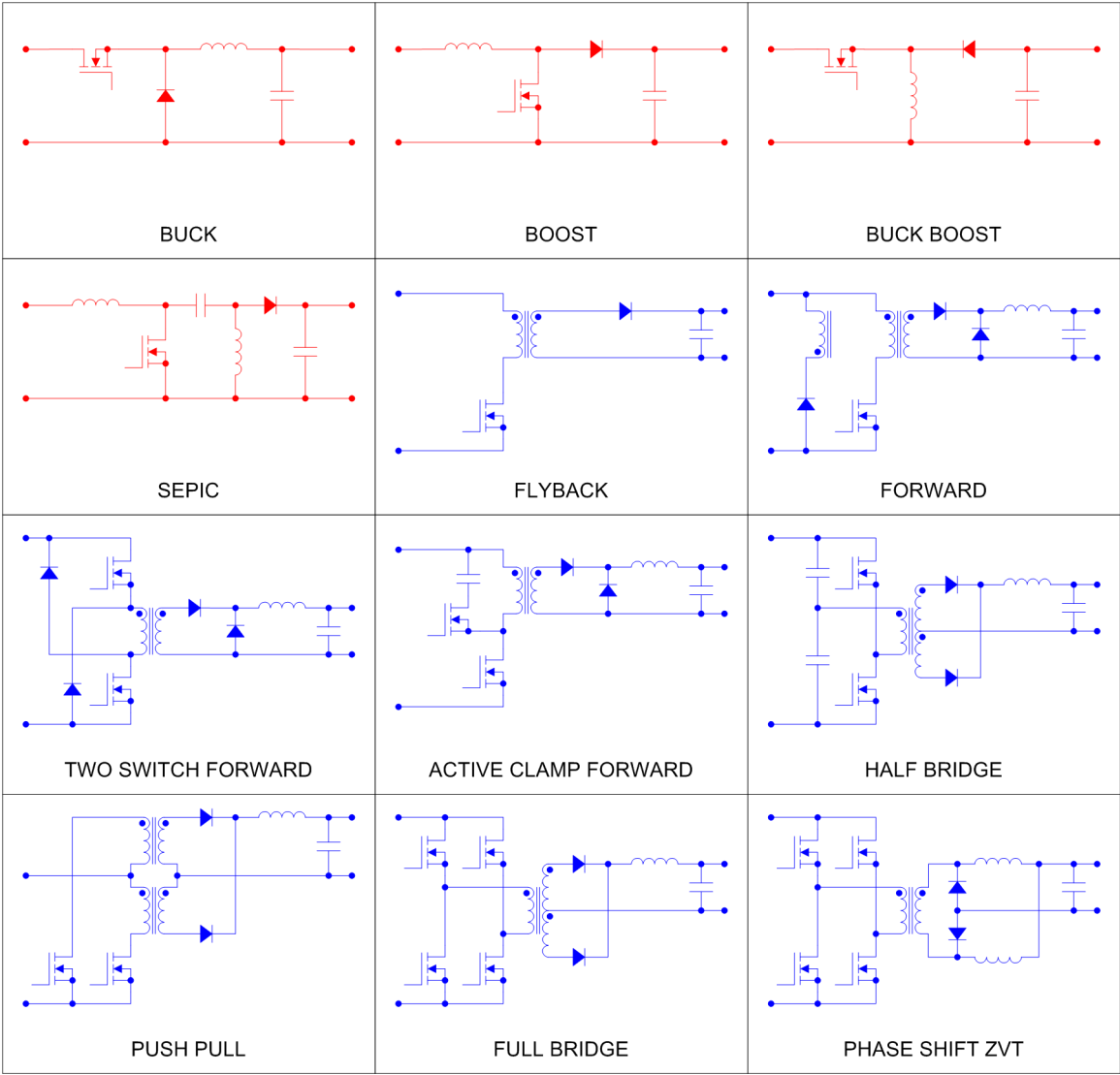
(CONTINUE)



BASIC OP AMP CONFIGURATIONS

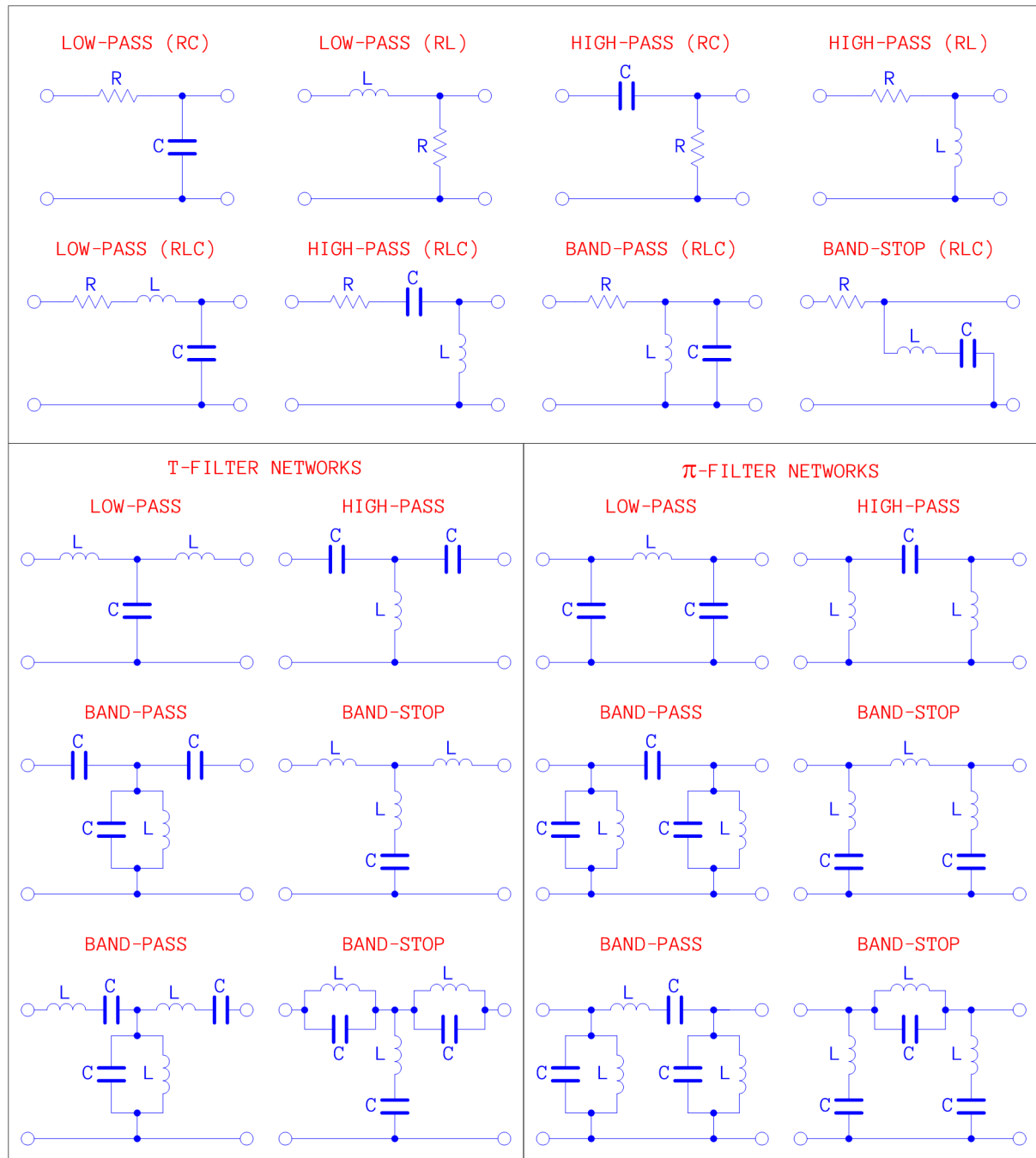
VOLTAGE COMPARATOR 	VOLTAGE FOLLOWER 
INVERTING AMPLIFIER 	NON-INVERTING AMPLIFIER 
INTEGRATOR 	DIFFERENTIATOR 
DIFFERENTIAL AMPLIFIER 	INVERTING SUMMING AMPLIFIER 

SWITCH-MODE POWER SUPPLY TOPOLOGIES

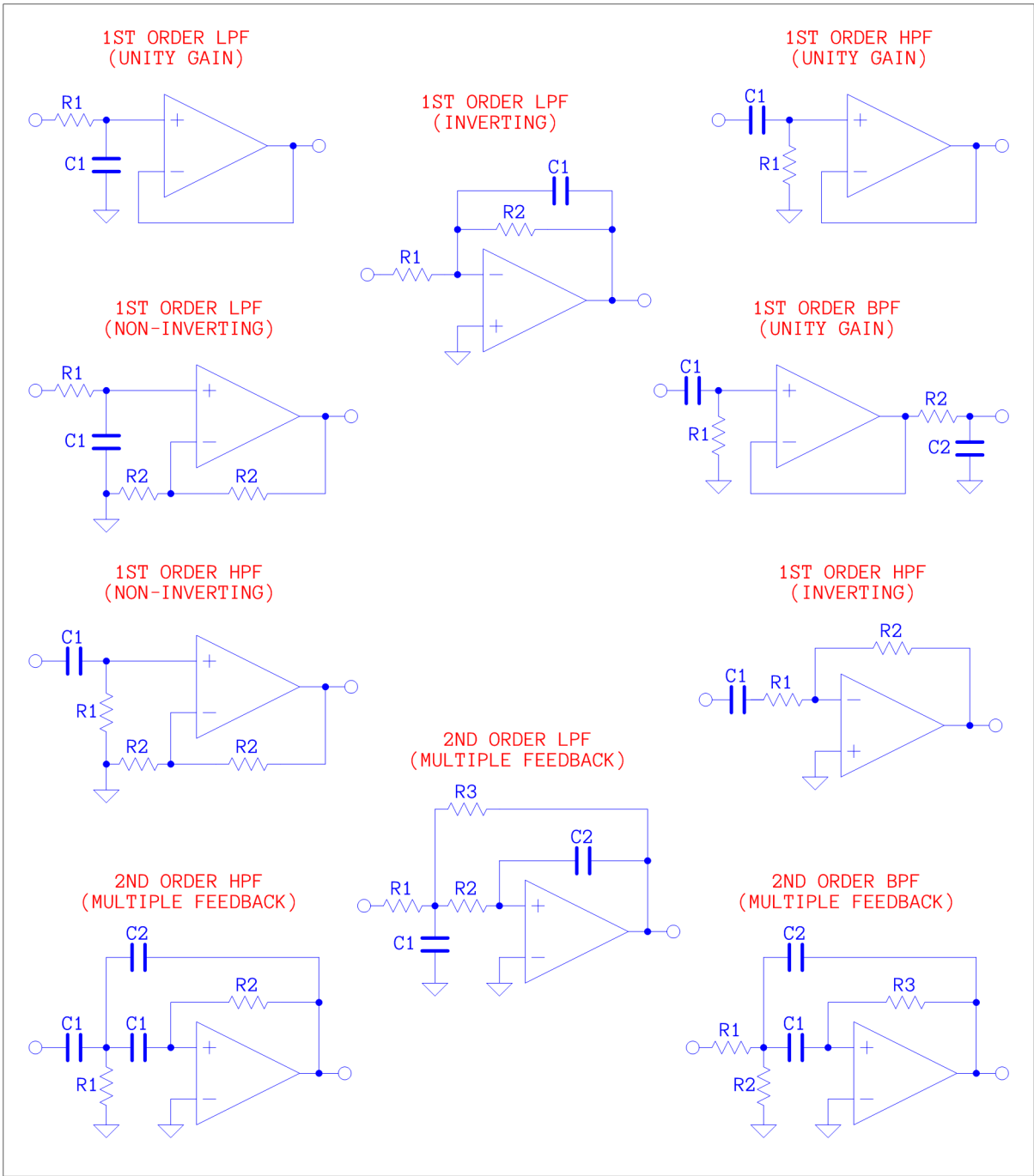


■ Non-isolated
■ Isolated

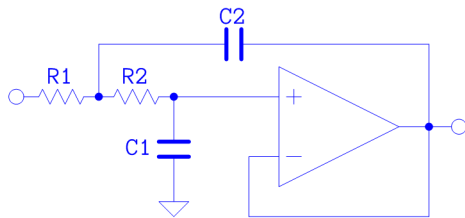
TYPES OF PASSIVE FILTER CIRCUITS



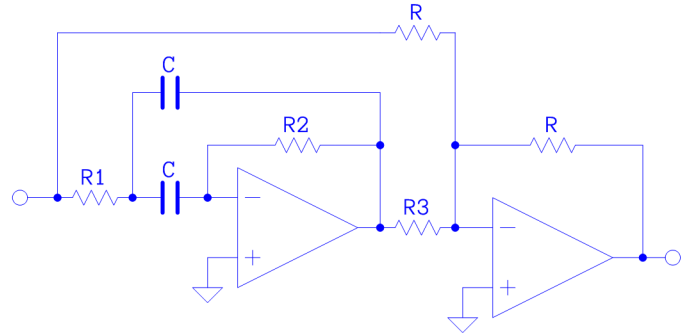
TYPES OF ACTIVE FILTER CIRCUITS



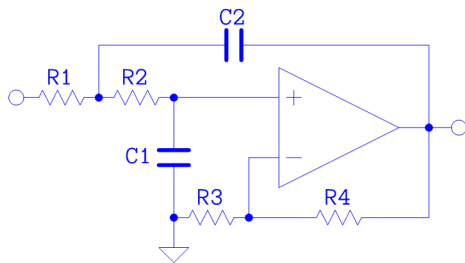
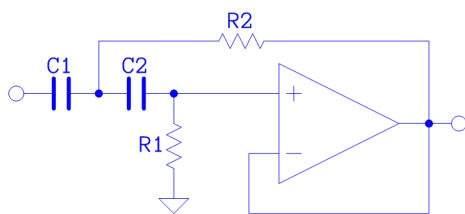
(CONTINUE)

2ND ORDER SALLEN-KEY LPF
(UNITY GAIN)

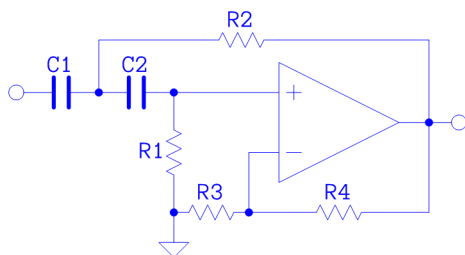
2ND ORDER ALL-PASS FILTER



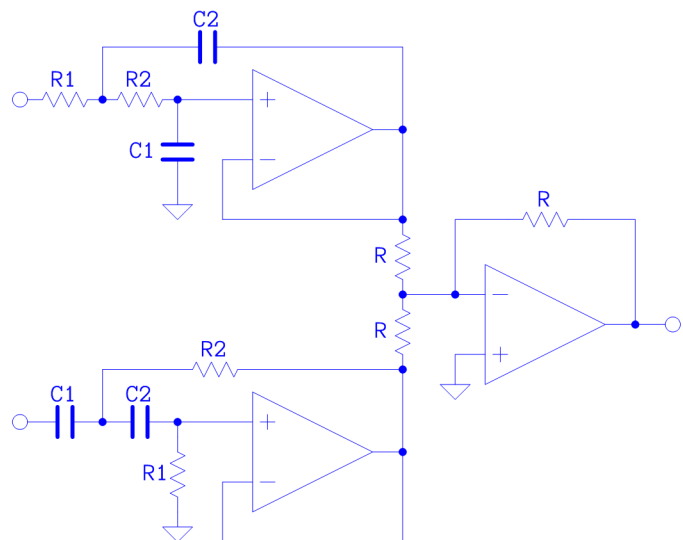
2ND ORDER SALLEN-KEY LPF

2ND ORDER SALLEN-KEY HPF
(UNITY GAIN)

2ND ORDER SALLEN-KEY HPF



BAND REJECT FILTER



Appendix E

ADVANCED TOPICS

DEVICES SUPPORTED BY JED2AHD.L.EXE	E-1
DOSBOX INSTALLATION & USAGE	E-3
MICROSOFT VIRTUAL PC INSTALLATION & USAGE	E-5

DEVICES SUPPORTED BY JED2AHD.L.EXE

E0310	EC16P8NC	F42VA12C	P14P4	P16V8R
E0320	EC16PE8	F473	P14P8	P16V8S
E0600	EC16RC4N	F506	P16C1	P16V8Z
E0600C	EC16RC8N	F507	P16H2	P16V8ZC
E0630	EC16RD4N	F529	P16H6	P16V8ZR
E0630C	EC16RD8N	F6001	P16H8	P16V8ZS
E0900	EC16RM4N	F6001C	P16HD8	P16VP8
E0900C	EC16TE6	F6002	P16L2	P16VP8C
E10P8	EC20EG8A	F6002C	P16L6	P16VP8R
E12P6	EC20EV8A	F82S100	P16L8	P16VP8S
E14P4	EC20EV8C	F82S103	P16L8C	P16Z8
E16P2	EC20G8M	F82S153	P16LD8	P16Z8AS
E16P8	EC20P8M	F82S162	P16LV8	P16Z8C
E16P8F	F100	F82S163	P16LV8C	P16Z8R
E16RP4	F103	F82S173	P16LV8R	P16Z8S
E16RP4F	F105	F839	P16LV8S	P18CV8
E16RP6	F151	F9800	P16N8	P18G8
E16RP6F	F153	F9804C0	P16P2	P18H4
E16RP8	F155	F9804C1	P16P6	P18L4
E16RP8F	F157	F9804C2	P16P8	P18N8
E1800	F159	F9804C4	P16R4	P18P4
E204	F161	F9806	P16R4C	P18P8
E204SIM	F162	F9808	P16R6	P18U8
E224	F163	FPGA2020	P16R6C	P18V10G
E224C	F167	H2010A40	P16R8	P18V8
E22VF10	F167C	H2010A44	P16R8C	P18V8Z
E22VF10C	F168	ISPGDS14	P16RA8	P19L8L
E242	F168C	ISPGDS18	P16RP4	P19L8R
E242C	F173	ISPGDS22	P16RP6	P19R4L
E5AC312	F173C	P10H8	P16RP8	P19R4R
E5AC324	F179	P10L8	P16SV8	P19R6L
EC12C4A	F179C	P10P8	P16V8	P19R6R
EC16C4A	F253	P12H10	P16V8AS	P19R8L
EC16ET6	F2605	P12H6	P16V8C	P19R8R
EC16LC4N	F2678	P12L10	P16V8HC	P20ARP10
EC16LC8N	F273	P12L6	P16V8HCC	P20ARP4
EC16LD4N	F30K12	P12P10	P16V8HD	P20ARP6
EC16LD8N	F30S16	P12P6	P16V8HDC	P20ARP8
EC16LM4N	F39V18	P14H4	P16V8HR	P20C1
EC16P4A	F405	P14H8	P16V8HRC	P20CG10
EC16P8C	F415	P14L4	P16V8HS	P20CG10C
EC16P8N	F42VA12	P14L8	P16V8HSC	P20G10
(continue...)				
P20G10C	P20SV8	P20XVFC	P29MA16C	PML501

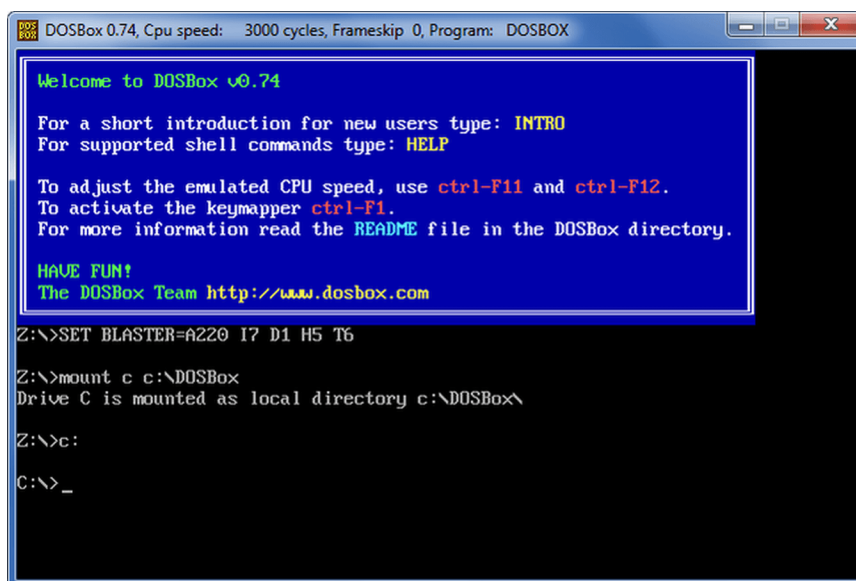
Appendix E

P20H2	P20V8	P20XVIN	P32VX10	PML601
P20H8	P20V8AS	P20XVINC	P32VX10A	RA10P4
P20L10	P20V8ASC	P22AP10	P32VX10C	RA10P8
P20L10C	P20V8C	P22CV10Z	P22CV10C	RA10R8
P20L2	P20V8CC	P22CV8	P330	RA11P4
P20L8	P20V8LCC	P22IP6	P331	RA11P8
P20L8C	P20V8R	P22RX8A	P332	RA11R8
P20L8FNC	P20V8RC	P22V10	P333	RA12P4
P20LV8	P20V8S	P22V10C	P335	RA12P8
P20LV8C	P20V8Z	P22V10G	P336	RA13P8
P20LV8CC	P20V8ZC	P22V10GC	P337	RA5P16
P20LV8LC	P20V8ZCC	P22V10GI	P338	RA5P8
P20LV8R	P20V8ZLC	P22V10I	P339	RA6P16
P20LV8RC	P20V8ZR	P22V10IC	P448	RA8P4
P20LV8S	P20V8ZRC	P22V10TI	P448C	RA8P8
P20LV8SC	P20V8ZS	P22VF10	P448Z	RA9P4
P20P2	P20V8ZSC	P22VP10	P464	RA9P8
P20P8	P20VP8	P22VP10C	P464C	RA9R8
P20R4	P20VP8C	P23S8	P464Z	RE8P4
P20R4C	P20VP8CC	P23SV8	P48N22	
P20R4FNC	P20VP8LC	P241	P5000	
P20R6	P20VP8R	P241SIM	P5016A	
P20R6C	P20VP8RC	P24L10	P5016S	
P20R6FNC	P20VP8S	P24R10	P5024A	
P20R8	P20VP8SC	P24V10	P5024S	
P20R8C	P20X10	P24V10C	P5032A	
P20R8FNC	P20X10C	P24V10R	P5032AC	
P20RA10	P20X4	P24V10S	P5032S	
P20RA10C	P20X4C	P2500	P5032SC	
P20RA10G	P20X8	P2500B	P508	
P20RA10N	P20X8C	P2500BC	P5100	
P20RP4	P20XRP10	P2500C	P6L16	
P20RP6	P20XRP4	P2500SIM	P750	
P20RP8	P20XRP6	P26CV12	P750B	
P20RS10	P20XRP8	P26V12	P750BC	
P20RS4	P20XV10	P29M16	P750C	
P20RS8	P20XV10C	P29M16C	P8L14	
P20S10	P20XVFB	P29MA16	PA202084	

* The more popular PLD models are highlighted in blue.

DOSBOX INSTALLATION & USAGE

DOSBox is a DOS emulator that can be installed and run within a variety of OSes such as Windows, OS/2, MAC OS X, and flavors of Unix like Solaris 10, BeOS, Fedora and Debian, etc. As of this writing, DOSBox is version 0.74:



Installation steps:

1. First, go to the DOSBox website (www.dosbox.com) and download the latest version. Run the installer and go with the default location suggested.
2. Next, you need to create a folder to mount as your DOSBox's C: drive. This is the folder that will store whatever you install inside DOSBox such as old DOS programs and games. For simplicity, create it in your C: root directory and name it as DOSBox (i.e. C:\DOSBox).
3. When you run DOSBox for the first time, you'll be presented with the above DOS window and the `Z:\>` prompt. To access the DOSBox folder and its content, you need to issue the command:
`mount c C:\DOSBox`

every time you run the emulator. To automate this step, go to **Start Menu** → **All Programs** → **DOSBox-0.74** → **Options** → **DOSBox 0.74 Options** and Notepad will open with a file named `dosbox-0.74.conf`. This is the configuration file for DOSBox.

4. Scroll down to the bottom of the file and add the following lines:

```
mount c C:\DOSBox
C:
```

and you're done! (see above figure.)

Usage:

Appendix E

1. By default, the installer will create a shortcut on the Desktop. Double-click on the DOSBox icon and the DOSBox window will appear. You can now proceed to run DOS commands or type and execute any DOS program at the command prompt.
2. Alternatively, you can execute DOSBox from the recent program list in the Start Menu if it is present, or go to [Start Menu](#) → [All Programs](#) → [DOSBox-0.74](#) → [DOSBox 0.74](#) to run it.

Pros:

1. Easy to install and run.
2. Small footprint.
3. Does not require additional virtual drive as medium.

Cons:

1. Environment is not customizable.
2. Limited to running DOS commands and programs only.
3. Cannot copy and paste text on screen.

Simply put, DOSBox is a quick fix terminal that allows you to work on legacy DOS programs like in the good old DOS era without all the bells and whistles. The only gripe I have with it is point 3 of the Cons, which is a real letdown to an otherwise neat little program. The way to work around this limitation is to use the redirect operator (>) to save the screen echo to a text log file:

```
jed2ahdl -i U1.JED -o U1.ABL -dev P16L8 > log.txt
```

and then open it using a text editor such as Notepad to copy the screen text out verbose.

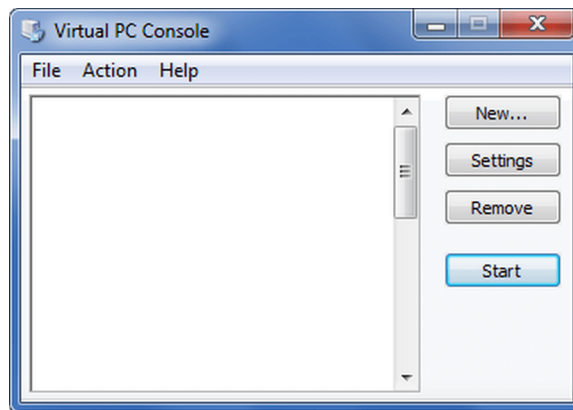
Note: Another DOS emulator quite similar to DOSBox is DOSEMU, but the popularity vote seems to swing in favor of DOSBox due to its portability to different OS platforms, as well as compatibility when running DOS programs and games.

MICROSOFT VIRTUAL PC INSTALLATION & USAGE

As the name implies, Microsoft's Virtual PC emulates a PC running inside a Windows environment. This means that you can create and run one or more virtual machines, each with its own operating system, on a single physical computer. To date, Microsoft has released a number of versions of its Virtual PC software package, the latest being Windows Virtual PC which, unlike its predecessors, supports only the Windows 7 host operating systems. For our purpose, I'll use the more versatile Virtual PC 2007 which supports the Windows XP host operating systems as well.

Installation steps:

1. Virtual PC 2007 is available free from Microsoft in 32- and 64-bit versions. Google to find the link and download the 32-bit version setup package, which is about 32MB in size.
2. Run the installer and select the default options by clicking [Next](#) and let the installation proceed to completion.
3. Launch the Virtual PC Console:



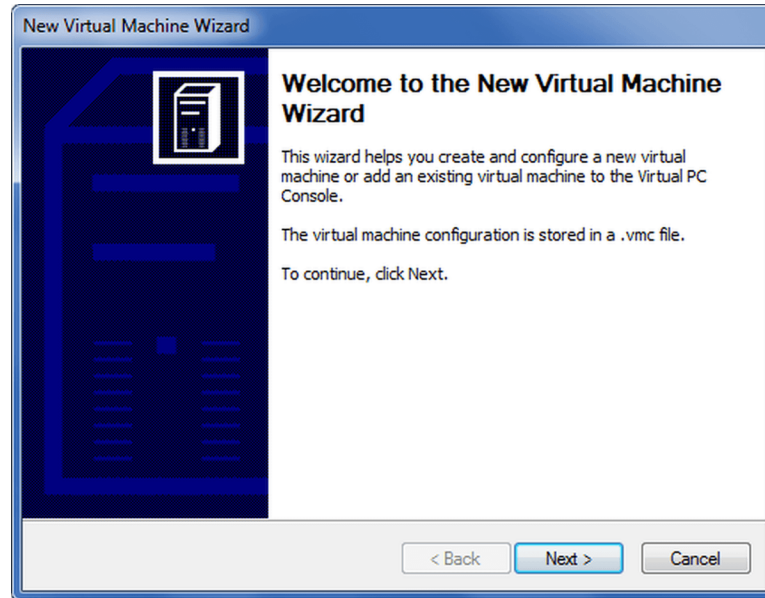
4. Currently there is no virtual machine (VM) in the list. We want to create a VM running MS-DOS version 6.22, but we're not going to do it the hard way.[†] Instead, go to the following website and download the MS-DOS 6.22 VM settings and hard disk image files (courtesy of JayRO):

<http://www.ohmancorp.com/refwin-virtualpc-vmnet622.asp>

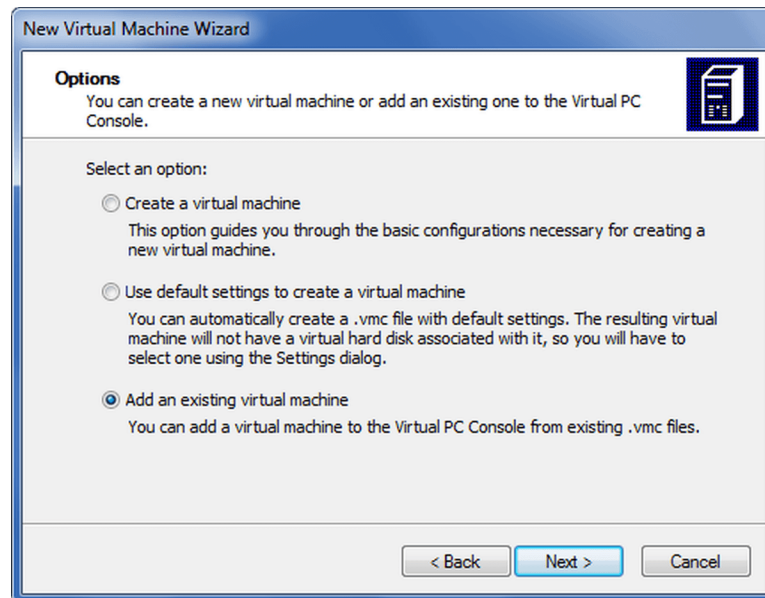
Unzip the content to a directory such as C:\My Virtual Machines\MSDOS622 and if you like, rename the long filenames to something shorter like MSDOS622, one with the HDD and one without to distinguish the larger hard disk image from the VM configuration (.vmc) file.

[†] I have a nagging feeling that many of my readers may not have a floppy drive available on their PCs, and even if some do have one by chance, the process of downloading the floppy images and transferring them to floppy disks if there's any on hand, or simply to buy a box of 10 just to use 3 for this purpose is both troublesome and wasteful, I wouldn't even think of asking my readers to do it!

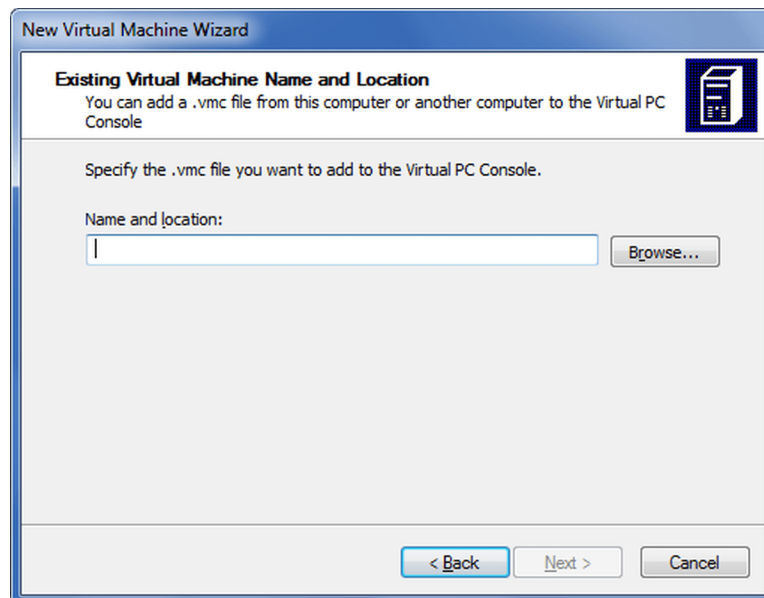
5. Now we're ready to add the MSDOS622 VM to our Virtual PC Console list. Click on the [New...](#) button on the Virtual PC Console. The New Virtual Machine Wizard appears:



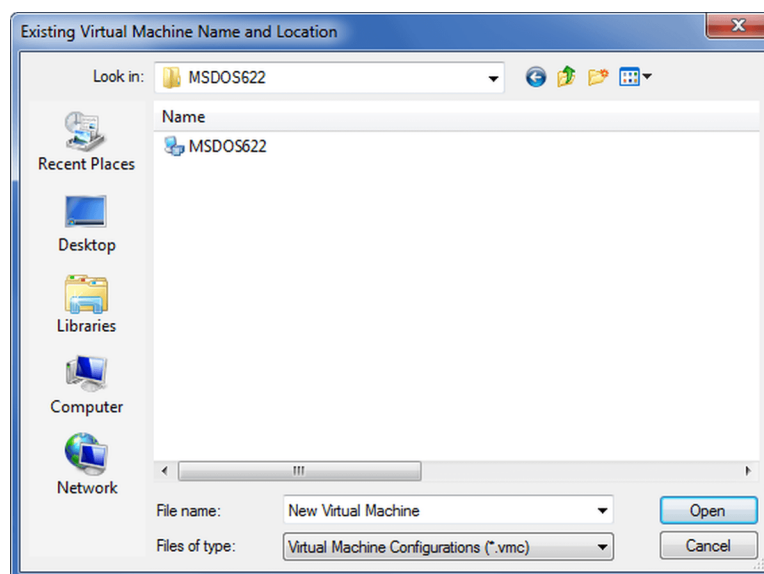
6. Click on [Next](#) and you'll be presented with three options: create a new VM, use default settings to create a VM, or add an existing VM. Select [Add an existing virtual machine](#), then click on [Next](#) again:



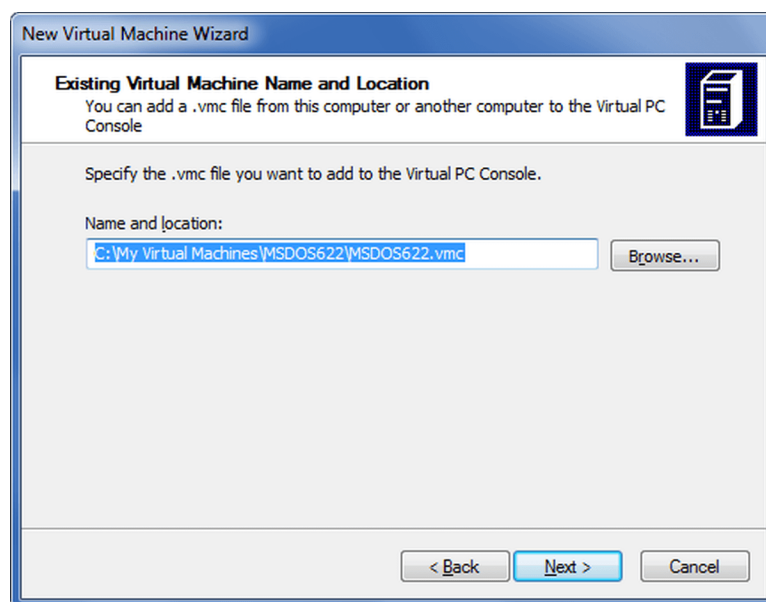
- You'll be prompted for the location of the VM's configuration file. Click on [Browse...](#) and then navigate to the sub-directory where MSDOS622.vmc resides:



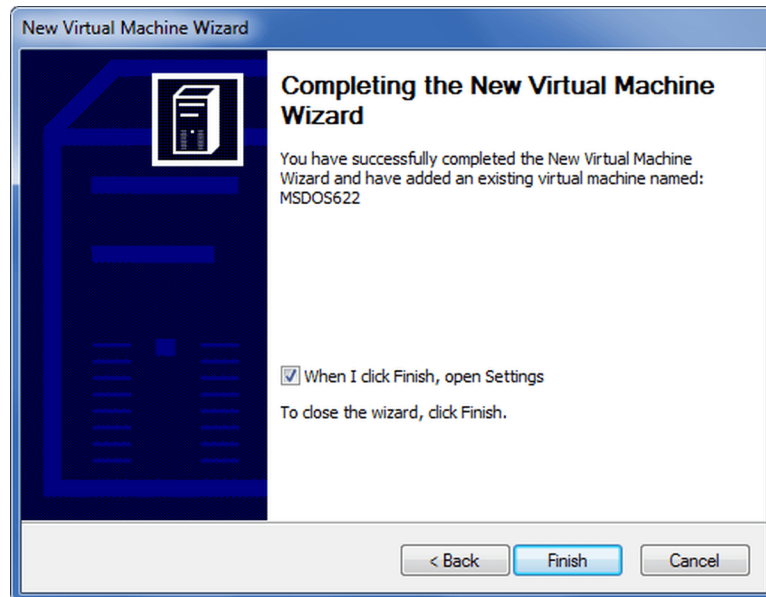
- Select MSDOS622 and click [OPEN](#):



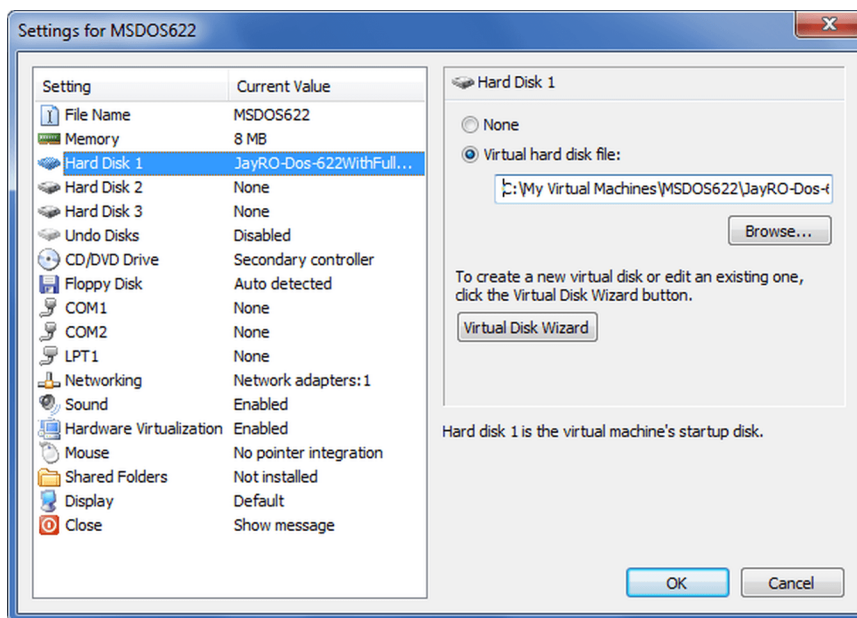
9. With the [Name and location](#) reflecting the correct path and filename, click [Next](#):



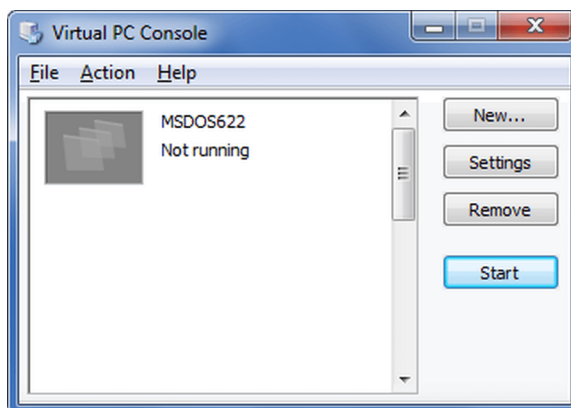
10. Finally, click on [Finish](#) and you're ready to go!



11. The [Settings for MSDOS622](#) dialog box appears, listing the available hardware resources and their current values. Notice that Hard Disk 1 reflects [JayRO-Dos-622WithFullNetworkingHDD](#). Click [Browse...](#) and select the hard disk image [MSDOS622HDD](#) which you've renamed. Then click on [OK](#).

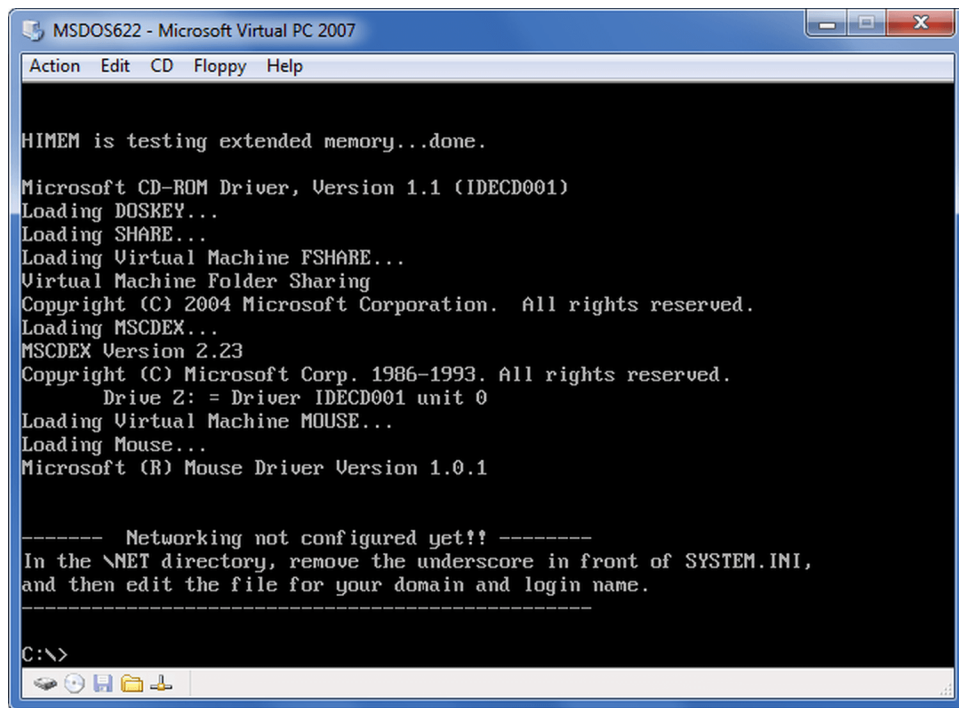


12. Now the Virtual PC Console contains one entry, which is MSDOS622:

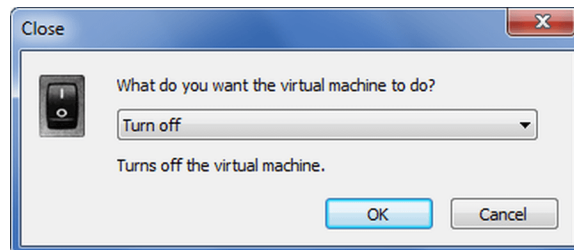


Select it and click [Start](#).

13. If everything runs well, you should see the following DOS window:



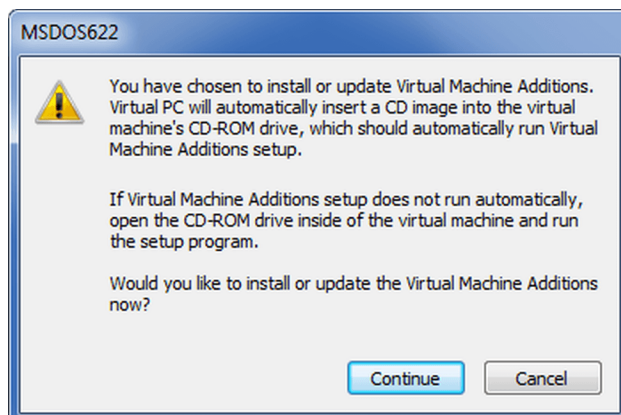
14. To shutdown the VM, select **Action** → **Close...** or press Right Alt+F4 and when the **Close** dialog box appears, check that Turn off is selected, then click on **OK**:



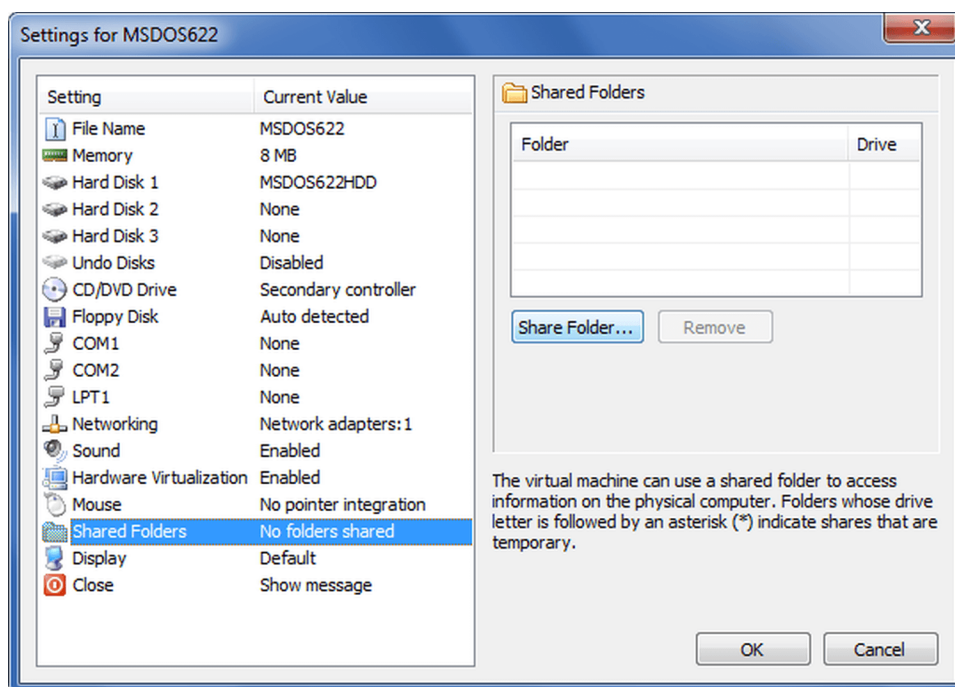
That's it!

Usage:

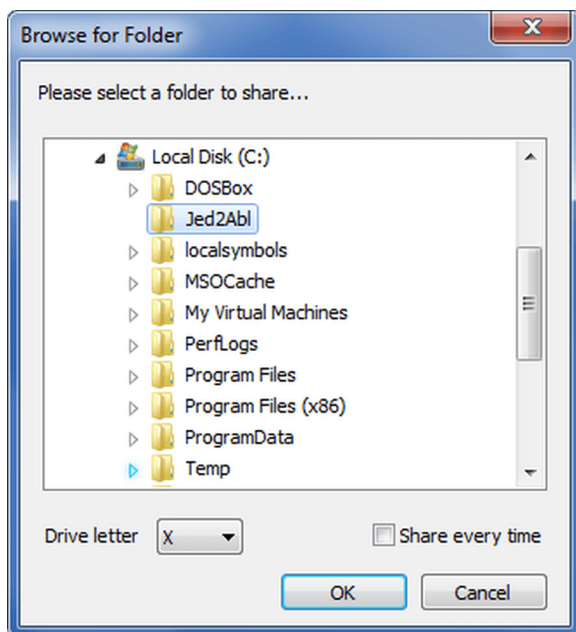
1. To use the JED2AHDLE.EXE program within the MS-DOS VM, you'll need to first download the JED2ABL.zip package as mentioned in [Chapter 5](#) under the [JEDEC De-Compiler](#) section and unzip it to a directory e.g. C:\JED2ABL.
2. To allow the MS-DOS VM share folders with the physical host computer, you need to perform a one-time activation by going to the menu and select [Action](#) → [Install or Update Virtual Machine Additions](#). Once this is done, you're ready to go.



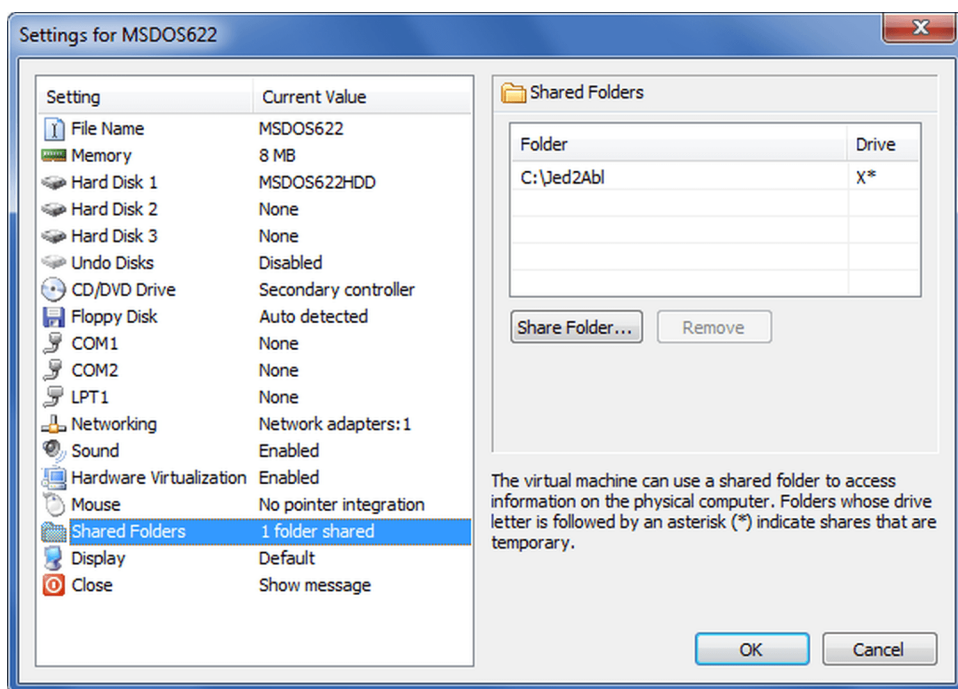
3. Assuming you've done just that, let's open the MS-DOS VM we've created earlier, select from menu [Edit](#) → [Settings](#) and click on Shared Folders. Notice that the current value is [No folders shared](#):



- Now click on the [Share Folder...](#) button and navigate to the [C:\JED2ABL](#) directory and select it. You can set the [Drive letter](#) or use the default assigned, which is [X](#) in this case:



- Now the [Settings](#) dialog box will show 1 folder shared under Shared Folders property and you can see that the [C:\Jed2Abl](#) folder is displayed with drive [X*](#) assigned to it. Click on [OK](#):



6. You can now change to drive **X** and execute JED2AHDL.EXE as shown:

```

MSDOS622 - Microsoft Virtual PC 2007
Action Edit CD Floppy Help
WINCUP~1 JED 1,269 10-17-14 7:11p
21 file(s) 1,077,490 bytes
2,147,450,880 bytes free

X:\>del u1.abl

X:\>jed2ahdl -i U1.JED -o U1.ABL -dev P16L8
JED2AHDL JEDEC to ABEL-HDL translator
ABEL 6.00 Copyright 1983-1994 Data I/O Corp. All Rights Reserved.

Input JEDEC file is: U1.JED.
Output AHDL file is: U1.ABL.
Device type is: P16L8.
Processing
    Reading Device Library
    Reading JEDEC Input File
    Extracting PLD Circuit Model
    Writing Output File
    DECLARATIONS Section
    EQUATIONS Section
    TEST_VECTORS Section
JED2AHDL complete. Time: 1 seconds.

X:\>

```

Whatever output files that JED2AHDL.EXE generated will be accessible via the physical C: drive as well since both real and VM shared the same folder, though with different drive assignments.

Note: By comparison, DOSBox is a more straightforward way to execute DOS programs than Virtual PC 2007 and that is certainly the recommendation if you intend to do only DOS and nothing else. The advantage that Virtual PC 2007 or other flavors of virtual machine software has to offer is the ability to run multiple VMs with different OSes at one time, so it's good to learn how to install and use them when future works require it.

Bibliographies

This author has taken great care to acknowledge sources of information, quotes, photos, illustrations, etc. either in the main texts, figures, footnotes, or in this bibliographies. Where possible, permissions are sought from known and traceable authors and originators, unless the works have been released into public domain sites or are freely-licensed and granted free right of use, such as Wikimedia Commons, or as declared in the originator's own website or hosted site. Photos and illustrations not listed here are owned and copyrighted by this author and should not be used without his expressed permission.

FIGURES

Figure	Source, Author, Originator
1.1a	Photo taken by author. Part of a PCB assembly.
1.1b	Source unknown
1.1c	ibid. 1.1a
1.1d	Source unknown
1.2b	Courtesy of Takaya Corporation
1.2c	Courtesy of Abi Electronics
1.2d	ibid. 1.1a
1.3	Artwork by author
1.4	ibid.
1.5	Photo taken by author. Part of a PCB assembly.
1.6	ibid. 1.1a
2.1	PicoChip product catalogue
2.2a-f	Various catalogs or unknown sources, with enhancement by the author, if necessary.
2.3	Artwork by author
2.4a-f	ibid. 2.2
2.5a-c	ibid.
2.6a-b	ibid.
2.7a-f	ibid.
2.8a-c	ibid.
2.9a-f	ibid.
2.10a-f	ibid.
2.11a-f	ibid.
2.12a-d	ibid.
2.13a-f	ibid.
2.14a-f	ibid.
2.15a	Kimmo Palosaari, Wikimedia Commons.
2.15b-f	ibid. 2.2
2.16a-c	ibid.

Bibliographies

- 2.17a-b *ibid.*
- 2.18 Photo taken by author. Part of a PCB assembly.
- 2.19 *ibid.*
- 2.20 *ibid.* 2.3
- 2.21 3M Company, Product catalog, Face masks and respirators.

- 3.1 Artwork by author
- 3.2 *ibid.*
- 3.3 Screen captured images. Source: Microsoft® Visio 2007 user interface.
- 3.4 *ibid.*
- 3.5 *ibid.* 3.1
- 3.6 Photo taken by author. ISA-bus NCR SCSI host adapter card.
- 3.7 *ibid.* 3.1
- 3.8 *ibid.*
- 3.9 *ibid.*
- 3.10 Adapted from IC data book specifications. Edited and enhanced by author.
- 3.11 *ibid.*
- 3.12 *ibid.* 3.1
- 3.13 *ibid.*
- 3.14 Photo taken by author. Motorola MVME 166-14A Single-board computer.
- 3.15 *ibid.* 3.1
- 3.16 *ibid.*
- 3.17 Photo taken by author. Part of a PCB assembly.
- 3.18 *ibid.*
- 3.19 *ibid.* 3.1
- 3.20 *ibid.*
- 3.21 *ibid.*
- 3.22 *ibid.*
- 3.23 *ibid.*
- 3.24 *ibid.*

- 4.1 Various catalogs or unknown sources, with enhancement or editing by author, if necessary.
- 4.2 *ibid.*
- 4.3 Courtesy of PACE Incorporated. Model: MBT-250.
- 4.4 Source: Wikimedia Commons. Contributed by various authors. http://commons.wikimedia.org/wiki/Logic_gates_unified_symbols.
- 4.5 Artwork by the author
- 4.6 *ibid.*
- 4.7 *ibid.*
- 4.8 *ibid.*
- 4.9 Screen captured image. Microsoft Visio 2007.
- 4.10 *ibid.* 4.5
- 4.11 *ibid.*
- 4.12 *ibid.*
- 4.13 Adapted from [The TTL Data Book for Design Engineers](#) by Texas Instruments.
- 4.14 *ibid.*
- 4.15 *ibid.* 4.5
- 4.16 *ibid.*

- 4.17 ibid.
- 4.18 ibid.
- 4.19 ibid.
- 4.20 ibid.
- 4.21 ibid.
- 4.22 ibid.
- 4.23 ibid.
- 4.24 ibid.
- 4.25 ibid.
- 4.26 ibid.
- 4.27 Adapted from page 9 of the SG6105 datasheet. Copyright © System General Corp. Version 2.4 (IR033.0011.B1). Redrawn and simplified for clarity by the author.
- 4.28 Courtesy of M-Tech. Model: KOB AP4450XA. Source: http://danyk.cz/s_atx_en.html.
- 4.29 ibid. 4.5
- 4.30 ibid.
- 4.31a-b Photo taken by author
- 4.32 ibid.

- 5.1 Artwork by author
- 5.2 Photo taken by author
- 5.3 ibid. 5.1
- 5.4 ibid.
- 5.5 ibid.
- 5.6 ibid.
- 5.7 ibid.
- 5.8 ibid.
- 5.9 ibid.
- 5.10 ibid.
- 5.11 Screen captured image by author
- 5.12 ibid. 5.1
- 5.13 ibid.
- 5.14 ibid.
- 5.15 ibid. 5.11

- 6.1 Screen captured image by author
- 6.2 ibid.
- 6.3 ibid.
- 6.4 Photo courtesy of Digi-Key
- 6.5 ibid. 6.1
- 6.6 Photo courtesy of Altium Limited.
- 6.7 ibid.
- 6.8 Screen capture image. Courtesy of Labcenter Electronics.
- 6.9 Adapted from Labcenter Electronics. Redrawn by author for enhancement.
- 6.10 ibid. 6.1

Bibliographies

INSERTS

Page	Source, Author, Originator
21	Artwork by author
22a	Source unknown
22b-e	Various catalogs or unknown sources, with enhancement or editing by author, if necessary.
24a-d	ibid.
25a-c	ibid.
27	ibid. 21
29	ibid.
30	Source unknown
31	ibid. 22b
33a-c	ibid.
34	ibid. 21
37a	MMI PAL Handbook, 3rd Edition, 1983.
37b	Source: Wikipedia
38a	Source: Computer History Museum
38b	ibid. 30
39	Courtesy of Sanchez Microelectronics, Inc.
41	ibid. 22b
57	Screen captured image
58a-b	ibid.
59a-b	ibid.
60-61	ibid.
63	ibid.
64a-b	ibid.
64c	ibid. 21
65a	ibid. 57
65b	ibid. 21
70	ibid.
71a-b	ibid.
73	ibid.
74a	ibid.
74b	ibid. 57
75-77	ibid. 21
80a-b	ibid.
81a,c	ibid.
81b	ibid. 22b
82a-c	ibid. 21
83a	ibid. 22b
83b	ibid. 21
85	ibid.
86a-b	ibid.
87a-b	ibid.
88	ibid. 57
89	ibid. 21

97	ibid.
98a-b	ibid.
99a-b	ibid.
100a-c	ibid.
101a	ibid. 22b
101b-c	ibid. 21
102a-b	ibid.
103a-c	ibid.
104a-b	ibid.
105a-b	ibid.
106a-c	ibid.
108a	ibid. 22b
108b	ibid. 21
109a	ibid. 22b
109b	ibid. 21
111a	ibid. 22b
111b-c	ibid. 21
112	ibid. 22b
113a-b	ibid. 21
116	Photo taken by author
123a-c	ibid.
124	ibid. 22b
126	ibid.
132	ibid. 57
133a-b	ibid. 21
134	ibid.
135a-c	ibid.
138a-b	ibid.
139-140	ibid.
142a-b	ibid.
143a-b	ibid.
144a-b	ibid.
145a-b	ibid.
146a-b	ibid.
147a-b	ibid.
148a-b	ibid.
150-151	ibid.
153	ibid.
154	ibid. 22b
155a-b	ibid. 21
156	ibid.
160a-b	ibid.
167	ibid.
169	ibid.
175	ibid. 22b
179	ibid.
182	ibid.
192	ibid. 57

Bibliographies

193a-b ibid.
194-195 ibid.
198a-b ibid.
200-202 ibid.
204-210 ibid.
212-216 ibid.
218-224 ibid.
226-228 ibid.
231-239 ibid.
241-247 ibid.
252 ibid. 21
253 ibid. 57

Note: Inserts are identified by the pages they appeared in, according to alphabetical order if there are multiple inserts per page.

TABLES

Table	Source, Author, Originator
2.1	Created by the author
2.2	ibid.
2.3	ibid.
4.1	ibid.
4.2	ibid.
5.1	ibid.
5.2	ibid.
5.3	Generated by Visio report, imported and formatted in Microsoft® Word by author.

APPENDICES

Ref.	Source, Author, Originator
A-1	Created by the author. Adapted from Color Codes for Resistors and Capacitors chart by Philips Electronic Components and Materials Division.
A-2	Source: Wikipedia
A-3	Artwork by author
A-4	Source: GM4PMK's SMD Codebook (partial listing as examples)
A-5	Source: TT Electronics Welwyn Components
A-6	Source: NovaCap Chip Marking System
A-7	Source: Lansdale Semiconductor Inc.
A-8	ibid.
A-9	ibid.
A-10	Logos are copyright and trademarks of their respective manufacturers.
A-11	Source: Texas Instruments (edited and enhanced by the author)
B-2	Copyright © CustomGuide. Used with permission.
B-3	ibid.
B-4	Samples from Visio stencil. Copyright © Microsoft Corporation.
B-5	ibid.
B-6	ibid.
B-7	ibid.
C-1	Source: Wikipedia
C-2	Artwork by author
C-3	Sources: online, various sources.
C-4	Adapted from data book. Edited and enhanced by the author for simplicity and clarity.
C-5	ibid.
C-6	ibid.
C-7	ibid.
C-8	ibid.
C-9	ibid.
C-10	ibid. C-2
C-11	Source: Unknown
D-1	Artwork by author
D-2	ibid.
D-3	ibid.
D-4	ibid.
D-5	ibid.
D-6	ibid.
D-7	ibid.
D-8	ibid.
D-9	ibid.
E-1	Source: Device listing from the JED2AHDH application program.
E-2	ibid.

Bibliographies

E-3	Text and screen capture images by author.
E-4	ibid.
E-5	ibid.
E-6	ibid.
E-7	ibid.
E-8	ibid.
E-9	ibid.
E-10	ibid.
E-11	ibid.
E-12	ibid.
E-13	ibid.

Chapters 1-6 cover arts are created and copyrighted © by the author.

FONTS

Fonts	Source, Author, Originator
Angelina	www.fontspace.com , created in 1995 by Anja Denali.
Data 70 LET	www.fontpalace.com , author unknown.
PCB	Created by author using Microsoft Wordart effect with superimposed PCB photo.

INDEX

A

- Accessibility, 32
- Advanced topics
 - JEDEC decompiler, 267
 - JEDEC, fusemap, 264
 - Layering
 - activate, 207
 - add, 206
 - assign shapes, 207
 - delete, 208
 - lock, 209
 - properties, 212
 - rename, 208
 - show, hide, 209
 - when to use, 205
- Report
 - bill of materials, 257
- Report
 - definition wizard, 258
- Report
 - exporting to Excel, 261
- Smartshapes, 219
 - multi-field shape, 239
 - multi-shape, 250
 - shape context menu, 255
 - shape data, 220
 - shape graphics, 224
 - shapesheets, 230
- Anecdotes
 - 6502 and Z80, 52
 - back injuries, 205
 - circuit simulation, 286
 - my personal story, 20
 - of PLDs and migraine, 263
 - prototyping, simulation, 160
 - resistor color codes, 35
 - technical drawing, 22

- the interns, 136
- transformers, 41
- vacuum tubes, valves, 46

B

- Basic tools
 - hand tools, 138
 - IC extractors, 139
 - magnifier lamp, 138
 - multimeter, 137
 - PACE workstation, 139
 - probe tips, 137
- Bill of materials, 33

C

- Case studies
 - ATX power supply, 189
 - infrared matrix sub-assembly, 23
 - ISA-bus SCSI host adapter, 80
 - layering a PCB, 210
 - MVME 166-14A SBC, 106
 - plasma touchscreen display, 21
- Commercial EDA tools
 - Altium Designer, 282
 - Proteus Design Suite, 284
- Components
 - capacitors, 37
 - crystals, oscillators, 45
 - datasheets, 62
 - DC-DC converters, 54
 - delay lines, 54
 - diodes, zeners, 46
 - fuses, 42
 - inductors, transformers, 40
 - integrated circuits, 49
 - missing reference designators, 56
 - prefixes, 59

Index

- relays, switches, 43
- resistors, networks, 35
- transistors, MOSFETs, 48
 - without markings, 55
- Conformal coatings, Types, 60

D

- Datasheets
 - Stephen M. Nolan, Jose M. Soltero, 62

E

- Equipment
 - flying probe test systems, 19
 - pizza-box system, 20

F

- Free EDA tools
 - DesignSpark PCB, 276
 - Digi-Key EDA Design Tools, 279
 - EasyEDA, 277
 - KiCAD, 274

I

- Integrated Circuits
 - analog, linear, 53
 - digital logic, 50
 - hybrids, 53
 - memories, 50
 - military grade, 53
 - peripherals, chipsets, 52
 - processors, CPUs, 52
 - Programmable Logic, 51

L

- Legacy EDA tools
 - DOS-based OrCAD, 280
- Legal
 - copyright issues, 25
 - David C. Musker, 25
 - non-disclosure agreement, 26
 - obsolescence, lack of support, 26

P

- PCB layout
 - ball-grid array, 116
 - bolts and nuts, 111
 - component layout symbols, 93
 - connectors, 112
 - discrete components, 93
 - grid references, title block, 82
 - integrated circuits, 97
 - metal can analog ICs, 126
 - necessity, purpose, 68
 - Page sizes, orientations, 81
 - PCB outline, 86
 - PCB-mount fixtures, 109
 - pin-grid array, 116
 - plastic leaded chip carrier, 118
 - populating the PCB, 102
 - quad flat pak, 118
 - scaling parts of a PCB, 131
 - simplified outline, 129
 - small outline ICs, 117
 - toggle switches, 115
 - transistors, heatsinks, 122
- Personal Protection Equipment, 61
- Pre-requisites, 18

S

- Schematic drawing
 - ANSI or IEC?, 141
 - base reference grid, 152
 - color vs monochrome, 142
 - common sense approach, 184
- Configurations
 - filters, 195
 - operational amplifiers, 193
 - power supplies, 194
- Elements
 - circuit layout, signal flow, 147
 - junctions, crosses, 149
 - page sizes, orientations, 146
 - power pins, decoupling caps, 149
 - reference designators, placement, 146
 - text, labels, 148
- hierarchical vs flat, 144

proper use of colors, 143

Special

gold finger, 198

magic carpet, 197

Strategy

analog, 192

digital, 175

Symbols

buses, 170

flip-flops, 155

integrated circuits, 164

logic gates, 152

opamps, 162

passive discretes, 157

power, ground, 158

relays, 161

semiconductors, 159

signals, labels, 174

tri-state logic, 154

T

Types of PCB, 17

V

Visio

aligning shapes, 78

arraying shapes, 78

developer mode, 72

distributing shapes, 78

guide lines, guide points, 73

initial settings, 70

keyboard shortcuts, 76

layering, 204

panning, scrolling, 77

rulers, grids, 73

sizing, positioning shapes, 79

snap, glue, 74

styles

fill, 75

font, 74

line, 75

text, 75

workspace, 69

zooming, 76



ABOUT THE AUTHOR

NG KENG TIONG is a Principal Engineer at Singapore Technologies (ST) Electronics Limited, a subsidiary of ST Engineering. Upon graduation from the Singapore Polytechnics, he signed up with the Republic of Singapore Air Force (RSAF) as an aircraft technician and worked in the E-2C Hawkeye repair bay, servicing the aircraft's avionics using automated test systems (CAT-IIID and RADCOM) and other in-house test equipment.

Upon invitation, he left the RSAF after his first contract and joined his present company, doing test program development and PCB diagnostics using Schlumberger S700 series testers. Since then, he has worked on other test platforms such as the Teradyne Spectrum™ 8800 series, the Westest-DATS/2000 test station, and some special-to-type-equipment (STTE) of similar nature. He also has experience in logic simulation using HHB Systems CADAT™ software and the CATS-10000™ hardware modeler, as well as Teradyne's LASAR™ simulator.

In the course of his work, he encountered many printed circuit boards and electronic modules without schematic diagrams or documentation. That started him on the journey of doing PCB reverse engineering, in part or total, to perform the necessary troubleshooting for repair. Over time, he has refined the skill into an art for which this book is all about.

OTHER BOOKS BY THE SAME AUTHOR

None—at least not yet...

This is the first book formally written and published by the author, though he has the intention of writing a few more to share his 30 years of experience in other engineering interests related to his work, such as PCB diagnostics and repair, test programming and fixture design, writing technical documentation, etc.

Since August of 2013 the author had requested to work on a part-time basis with his current company, partly to spend more time with his aging mother and to explore other avenues of work from home, of which writing this book is one of them. He hopes that this first attempt at book writing will have sufficient measure of success to enable him to quit his day time job completely, so he can focus on writing a few more to benefit the electronics engineering community.